intel

82786 Graphics Coprocessor User's Manual



Order Number: 231933-003

LITERATURE

To order Intel Literature write or call:

INTEL LITERATURE SALES
P.O. BOX 58130
SANTA CLARA, CA 95052-8130

TOLL FREE NUMBER:
(800) 548-4725*

1988 HANDBOOKS

Product line handbooks contain data sheets, application notes, article reprints and other design information.

TITLE	LITERATURE ORDER NUMBER
COMPLETE SET OF 8 HANDBOOKS Save $50.00 off the retail price of $175.00. (Price applicable to U.S. and Canadian shipments only)	**231003**
AUTOMOTIVE HANDBOOK, 1200 pages (Not included in handbook set)	231792
COMPONENTS QUALITY/RELIABILITY HANDBOOK, 288 pages (Available in July)	210997
EMBEDDED CONTROLLER HANDBOOK, 2016 pages (2 volume set)	210918
MEMORY COMPONENTS HANDBOOK, 528 pages	210830
MEMORY COMPONENTS HANDBOOK SUPPLEMENT, 256 pages (Available in July)	230663
MICROCOMMUNICATIONS HANDBOOK, 1648 pages	231658
MICROPROCESSOR AND PERIPHERAL HANDBOOK, 2224 pages (2 volume set)	230843
MILITARY HANDBOOK, 1776 pages (Not included in handbook set)	210461
OEM BOARDS AND SYSTEMS HANDBOOK, 880 pages	280407
PROGRAMMABLE LOGIC HANDBOOK, 448 pages	296083
SYSTEMS QUALITY/RELIABILITY HANDBOOK, 160 pages	231762
PRODUCT GUIDE (No charge) Overview of Intel's complete product lines	210846
DEVELOPMENT TOOLS CATALOG (No charge)	280199
INTEL PACKAGING OUTLINES AND DIMENSIONS (No charge) Packaging types, number of leads, etc.	231369
LITERATURE PRICE LIST (No charge) List of Intel Literature	210620

For U.S. and Canadian literature pricing, call or write Intel Literature Sales. In Europe and other international locations, please contact your local Intel Sales Office or Distributor for literature prices.

*Good in the U.S. and Canada.

Intel Literature update Service

Get Intel's Latest Technical Literature, Automatically!

Exclusive, Intel Literature Update Service

Take advantage of Intel's year-long, low cost Literature Update Service and you will receive your first package of information followed by automatic quarterly updates on all the latest product and service news from Intel.

Choose one or all five product categories update

Each product category update listed below covers in depth, all the latest Handbooks, Data Sheets, Application Notes, Reliability Reports, Errata Reports, Article Reprints, Promotional Offers, Brochures, Benchmark Reports, Technical Papers and much more. . .

1. Microprocessors

Product line handbooks on Microprocessors, Embedded Controllers and Component Quality/Reliability, *Plus,* the Product Guide, Literature Guide, Packaging Information and 3 quarterly updates. $70.00 Order Number: **555110**

2. Peripherals

Product line handbooks on Peripherals, Microcommunications, Embedded Controllers, and Component Quality/Reliability, *Plus,* the Product Guide, Literature Guide, Packaging Information and 3 quarterly updates. $50.00 Order Number: **555111**

3. Memories

Product line handbooks on Memory Components, Programmable Logic and Components Quality/Reliability, *Plus,* the Product Guide, Literature Guide, Packaging Information and 3 quarterly updates. $50.00 Order Number: **555112**

4. OEM Boards and Systems

Product line handbooks on OEM Boards & Systems, Systems Quality/Reliability, *Plus,* the Product Guide, Literature Guide, Packaging Information and 3 quarterly updates. $50.00 Order Number: **555113**

5. Software

Product line handbooks on Systems Quality/Reliability, Development Tools Catalog, *Plus,* the Product Guide, Literature Guide, Packaging Information and 3 quarterly updates. $35.00 Order Number: **555114**

To subscribe, rush the Literature Order Form in this handbook, or call today, toll free (800) 548-4725.*

Subscribe by March 31, 1988 and receive a valuable free gift.

The charge for this service covers our printing, postage and handling cost only.

Please note: Product manuals are not included in this offer.

Customers outside the U.S. and Canada should order directly from the U.S.

Offer expires 12/31/88.

*Good in the U.S. and Canada.

intel

intel

LITERATURE SALES ORDER FORM

NAME: _____

COMPANY: _____

ADDRESS: _____

CITY: _____ STATE: _____ ZIP: _____

COUNTRY: _____

PHONE NO.: ( ) _____

ORDER NO.	TITLE	QTY.	PRICE	TOTAL

Subtotal _____

Must Add Your Local Sales Tax _____

Must add appropriate postage to subtotal (10% U.S. and Canada, 20% all other).

Postage _____

Total _____

Pay by Visa, MasterCard, American Express, Check, Money Order, or company purchase order payable to Intel Literature Sales. Allow 2-4 weeks for delivery.

☐ Visa ☐ MasterCard ☐ American Express Expiration Date _____

Account No. _____

Signature _____

Mail To: Intel Literature Sales
P.O. Box 58130
Santa Clara, CA 95052-8130

International Customers outside the U.S. and Canada should contact their local Intel Sales Office or Distributor listed in the back of most Intel literature.

Call Toll Free: (800) 548-4725 for phone orders

Prices good until 12/31/88.

Source HB

CG/LSOF/062188

CUSTOMER SUPPORT

CUSTOMER SUPPORT

Customer Support is Intel's complete support service that provides Intel customers with hardware support, software support, customer training, and consulting services. For more information contact your local sales offices.

After a customer purchases any system hardware or software product, service and support become major factors in determining whether that product will continue to meet a customer's expectations. Such support requires an international support organization and a breadth of programs to meet a variety of customer needs. As you might expect, Intel's customer support is quite extensive. It includes factory repair services and worldwide field service offices providing hardware repair services, software support services, customer training classes, and consulting services.

HARDWARE SUPPORT SERVICES

Intel is committed to providing an international service support package through a wide variety of service offerings available from Intel Hardware Support.

SOFTWARE SUPPORT SERVICES

Intel's software support consists of two levels of contracts. Standard support includes TIPS (Technical Information Phone Service), updates and subscription service (product-specific troubleshooting guides and COMMENTS Magazine). Basic support includes updates and the subscription service. Contracts are sold in environments which represent product groupings (i.e., iRMX® environment).

CONSULTING SERVICES

Intel provides field systems engineering services for any phase of your development or support effort. You can use our systems engineers in a variety of ways ranging from assistance in using a new product, developing an application, personalizing training, and customizing or tailoring an Intel product to providing technical and management consulting. Systems Engineers are well versed in technical areas such as microcommunications, real-time applications, embedded microcontrollers, and network services. You know your application needs; we know our products. Working together we can help you get a successful product to market in the least possible time.

CUSTOMER TRAINING

Intel offers a wide range of instructional programs covering various aspects of system design and implementation. In just three to ten days a limited number of individuals learn more in a single workshop than in weeks of self-study. For optimum convenience, workshops are scheduled regularly at Training Centers worldwide or we can take our workshops to you for on-site instruction. Covering a wide variety of topics, Intel's major course categories include: architecture and assembly language, programming and operating systems, BITBUS™ and LAN applications.



82786 GRAPHICS COPROCESSOR USER'S MANUAL

1988

Intel Corporation makes no warranty for the use of its products and assumes no responsibility for any errors which may appear in this document nor does it make a commitment to update the information contained herein.

Intel retains the right to make changes to these specifications at any time, without notice.

Contact your local sales office to obtain the latest specifications before placing your order.

The following are trademarks of Intel Corporation and may only be used to identify Intel Products:

Above, BITBUS, COMMputer, CREDIT, Data Pipeline, ETOX, FASTPATH, Genius, i, i, ICE, iCEL, iCS, iDBP, iDIS, I²ICE, iLBX, i_m, iMDDX, iMMX, Inboard, Insite, Intel, intel, Intel376, Intel386, Intel486, intelBOS, Intel Certified, Intelevision, inteligent Identifier, inteligent Programming, Intellec, Intellink, iOSP, iPDS, iPSC, iRMK, iRMX, iSBC, iSBX, iSDM, iSXM, KEPROM, Library Manager, MAPNET, MCS, Megachassis, MICROMAINFRAME, MULTIBUS, MULTICHANNEL, MULTIMODULE, ONCE, OpenNET, OTP, PC BUBBLE, Plug-A-Bubble, PROMPT, Promware, QUEST, QueX, Quick-Erase, Quick-Pulse Programming, Ripplemode, RMX/80, RUPI, Seamless, SLD, SugarCube, UPI, and VLSiCEL, and the combination of ICE, iCS, iRMX, iSBC, iSBX, iSXM, MCS, or UPI and a numerical suffix, 4-SITE, 376, 386, 486.

MDS is an ordering code only and is not used as a product name or trademark. MDS® is a registered trademark of Mohawk Data Sciences Corporation.

*MULTIBUS is a patented Intel bus.

Additional copies of this manual or other Intel literature may be obtained from:

Intel Corporation
Literature Sales
P.O. Box 58130
Santa Clara, CA 95052-8130

©INTEL CORPORATION 1988

CG-3/10/88

TABLE OF CONTENTS

CHAPTER 1 **Page**

82786 OVERVIEW

1.0 Revision Information 1-2
1.1 Architecture 1-3
1.1.1 Graphics Processor (GP) 1-3
1.1.2 Display Processor/CRT Controller (DP) 1-3
1.1.3 Display Processor (DP) Master and Slave Modes 1-4
1.1.4 Bus Interface Unit (BIU) 1-4
1.1.5 Memory Structure 1-5
1.1.6 Memory Access and Arbitration 1-5
1.1.7 Master and Slave Memory Access Interfaces 1-5
1.1.7.1 Master Mode Interface 1-6
1.1.7.2 Slave Interface 1-6
1.1.8 Internal Registers 1-6
1.2 System Configurations 1-6

CHAPTER 2

GRAPHICS PROCESSOR OVERVIEW

2.1 Graphics Concepts 2-1
2.1.1 Bitmaps 2-1
2.1.2 Bitmap Coordinates 2-2
2.1.3 Pixels 2-2
2.1.4 Calculating the Effective Address (EA) 2-3
2.1.5 Windows 2-4
2.2 Graphics Processor (GP) Registers 2-4
2.2.1 Internal Graphics Processor (GP) Registers 2-4
2.2.1.1 Graphics Processor Status Register (GSTAT) 2-5
2.2.1.2 Graphics Processor Instruction Pointer 2-5
2.2.2 Graphics Processor (GP) Control Registers 2-6
2.2.3 Graphics Processor (GP) Context Registers 2-7
2.3 Command Execution and Format 2-8
2.3.1 Graphics Processor Command Block (GCMB) Format 2-9
2.3.2 Poll State 2-10
2.3.3 RESET and Initialization 2-10
2.3.4 Exception Handling 2-12
2.4 Graphics Processor (GP) Commands 2-13
2.5 Nondrawing Commands 2-13
2.6 Drawing Control Commands 2-14
2.6.1 Attributes Associated with Drawing Commands 2-14
2.6.2 Color Bit Mask 2-15
2.6.3 Logical Operations 2-15
2.6.4 Clipping Rectangle 2-16
2.6.5 Pick Mode 2-16

Page

2.6.6 Foreground and Background Colors 2-17
2.6.7 Transparent or Opaque Mode 2-17
2.6.8 Pattern Mask 2-17
2.7 Geometric Commands 2-18
2.8 Bit Block Transfer (Bitblt) Commands 2-18
2.9 Character Command 2-18
2.9.1 Character Font Support 2-18
2.9.2 Character Storage 2-19
2.10 Command Descriptions 2-23
2.10.1 ABS_MOV – Move 2-24
2.10.2 ARC – Draw Arc 2-25
2.10.3 BIT_BLT – Bit Block Transfer 2-27
2.10.4 BIT_BLT_M – Bit Block Transfer Between Bitmaps 2-29
2.10.5 CALL – Call Subroutine 2-31
2.10.6 CHAR – Draw Character String 2-32
2.10.7 CIRCLE – Draw Circle 2-38
2.10.8 DEF_BITMAP – Define Bitmap 2-39
2.10.9 DEF_CHAR_ORIENT – Define Character Orientation 2-41
2.10.10 DEF_CHAR_SET – Define Character Set 2-43
2.10.11 DEF_SPACE – Define Space 2-48
2.10.12 DEF_CLIP_RECT – Define Clipping Rectangle 2-49
2.10.13 DEF_COLORS – Define Colors 2-51
2.10.14 DEF_LOGICAL_OP – Define Logical Operation 2-52
2.10.15 DEF_TEXTURE – Define Texture 2-55
2.10.16 DUMP_REG – Dump Register 2-56
2.10.17 EBITBLT—Expand 1bpp Source to Currently Defined Bpp 2-58
2.10.18 ENTER_PICK – Enter PICK Mode 2-60
2.10.19 EXIT_PICK – Exit PICK Mode 2-61
2.10.20 HALT – Stops Command Execution 2-62
2.10.21 INCR_POINT – Draw Series of Incremental Points 2-63
2.10.22 INTR_GEN – Generate Interrupt 2-66
2.10.23 LINE – Draw Line 2-67
2.10.24 LINK – Link Next Command 2-69
2.10.25 LOAD_REG – Load Register 2-70
2.10.26 NOP – No Operation 2-71
2.10.27 POINT – Draw Point 2-72
2.10.28 POLYGON – Draw Polygon 2-74
2.10.29 POLYLINE – Draw Polyline 2-76
2.10.30 RECT – Draw Rectangle 2-78
2.10.31 REL_MOV – Relative Move 2-80
2.10.32 RETURN – Return from Subroutine 2-81
2.10.33 SCAN_LINES – Draw Series of Horizontal Lines 2-82

CHAPTER 3 **Page**
DISPLAY PROCESSOR OVERVIEW
3.1 Display Processor Operation 3-1
3.1.1 Bitmap Organization 3-1
3.1.2 IBM PC Bitmap Format Support 3-2
3.1.3 Window Display Format 3-4
3.1.3.1 Strip Descriptors 3-4
3.1.3.2 Strip Descriptor Format 3-5
3.1.4 Cursor 3-9
3.1.5 Bus Bandwidth Requirements 3-10
3.2 Video Interface 3-10
3.2.1 CRT Controller 3-11
3.2.2 Video Rates 3-12
3.2.3 HSync and VSync Multiplex Window Status 3-13
3.2.4 Zoom Support 3-13
3.2.5 VRAM Support 3-15
3.2.5.1 Sample VRAM Design 3-15
3.2.5.2 Hardware Overlays 3-17
3.2.5.3 Initiating VRAM Mode and Functions 3-17
3.2.6 Extended 82786 System Configurations 3-18
3.3 Display Processor Registers 3-19
3.3.1 Display Processor Internal Registers 3-19
3.3.1.1 DPStatus Register and Exception Handling 3-20
3.3.1.2 Interrupts 3-21
3.3.2 Display Control Block Registers 3-21
3.3.3 Pad Registers 3-21
3.4 Display Command Synchronization 3-21
3.5 Command Execution 3-27
3.6 Display Processor Register Commands 3-29
3.6.1 Load Register (LD RG) 3-29
3.6.2 Load All (LD ALL) 3-29
3.6.3 Dump Register (DMP RG) 3-30
3.6.4 Dump All (DMP ALL) 3-30
3.6.5 Loop Command 3-30
3.6.5.1 Updating Display Processor Registers with Loop Mode 3-31
3.6.5.2 Suggested Mode of Operation 3-32
3.6.6 Write Protect Bit 3-32
3.7 Initialization 3-33
3.8 Video Data Signature Analyzer 3-33
3.8.1 Invoking Test Mode 3-34
3.8.2 Operation of the Signature Analyzer 3-34

CHAPTER 4 **Page**
BUS INTERFACE UNIT OVERVIEW
4.1 Memory Structure and Internal Registers 4-2
4.2 Internal Registers 4-2
4.2.1 Relocation Register 4-3
4.2.2 Control Register 4-5
4.2.3 DRAM/VRAM Refresh Control Register 4-6
4.2.4 DRAM/VRAM Control Register 4-8
4.2.5 Display Priority Register 4-9
4.2.6 Graphics Priority Register 4-9
4.2.7 External Priority Register 4-10
4.3 Bus Cycle Arbitration 4-11
4.3.1 Priority Levels 4-11
4.3.2 Priority Exceptions 4-12
4.4 Master and Slave Interfaces 4-13
4.4.1 Master Mode Interface 4-13
4.4.1.1 Initiating Master Mode Interface 4-13
4.4.1.2 Retaining Control of the System Bus 4-14
4.4.2 Slave Interface 4-14
4.4.2.1 Initiating a Slave Request 4-15
4.4.2.2 8-Bit and 16-Bit Interfaces 4-15
4.4.2.3 Slave Interface Byte Access to Memory 4-17
4.4.2.4 Slave Enable (SEN) as Ready Indication 4-17
4.4.2.5 Accessing Internal Registers & Graphics Memory 4-17
4.4.2.6 Command Lockout 4-18
4.5 System Bus Interface 4-18
4.5.1 Synchronous 80286 Interface 4-19
4.5.2 80186 Synchronous Interface 4-22
4.5.3 Asynchronous Interface 4-23
4.6 Performance 4-25
4.7 RESET Conditions 4-25
4.7.1 Special Pins 4-25
4.7.2 Initialization 4-25

CHAPTER 5
GRAPHICS MEMORY
5.1 DRAM/VRAM Configurations 5-1
5.1.1 VRAM Considerations 5-5
5.1.2 Considerations for ×1 Memory Devices 5-5
5.1.3 Illegal DRAM/VRAM Control Register Values 5-7
5.1.4 Data Line Connection 5-8
5.2 DRAM/VRAM Cycle Types 5-8
5.3 Graphics Memory Refresh 5-10
5.3.1 Refresh Latency 5-10
5.3.2 Refresh after RESET 5-12

Page

5.4 Configuring and Accessing Graphics Memory 5-12
5.5 Memory Map 5-13
5.5.1 Mapping the Internal Registers 5-14
5.5.2 Alternatives for Graphics Memory Mapping 5-15

CHAPTER 6
VIDEO INTERFACE
6.1 CRT Interfaces 6-2
6.1.1 CRTs with TTL-Level Inputs 6-2
6.1.2 CRTs with Analog Inputs 6-3
6.2 Using a Color Lookup Table/Video Palette Ram 6-5
6.3 Window Status Signals 6-7
6.3.1 Controlling the Cursor 6-7
6.3.2 Window Status Bits Segment Lookup Table for Multiple Windows 6-9
6.4 High Resolutions with Standard DRAMs 6-10
6.4.1 External Logic Requirements 6-10
6.4.2 Software Changes Required for High Resolution 6-11
6.4.3 Cursor Control 6-12
6.4.4 Zoom in Accelerated Modes 6-12
6.4.5 Examples of High Resolution Configurations 6-12
6.5 Greater Resolution with Multiple 82786s 6-14
6.5.1 An Interlaced Display with Two or More 82786s 6-16
6.5.2 Bitmap Configurations with Dual 82786s 6-16
6.6 Video RAM (VRAM) Interface 6-18
6.7 External Character ROM 6-18
6.8 Combining the 82786 with Other Video Sources 6-20
6.9 Other Types of Displays and Printers 6-23
6.9.1 Pixel Clock Rate 6-23
6.9.2 Partial Display Updates 6-23
6.9.3 Pixel Address Generation 6-24
6.9.4 Super High Resolution 6-24
6.10 Calculating Video Parameters 6-25
6.10.1 Application Parameters 6-26
6.10.2 Monitor Parameters 6-27
6.10.3 Video Interface Parameters 6-29
6.10.4 Sample Video Parameter Calculations 6-30
6.11 A Spreadsheet for Calculating Video Parameters 6-34

APPENDIX A
TEST MODES

APPENDIX B
88-PIN GRID ARRAY

APPENDIX C
82786 COMMANDS

Figures

Figure	Title	Page
1-1	82786 Functional Overview	1-1
1-2	Sequential Ordering Replaces Traditional Bit Plane Model	1-4
1-3	82786 128-Byte Internal Register Block	1-8
1-4	Low-End Low-Priced Personal Computer	1-9
1-5	Multi-Tasking Office Workstation	1-9
1-6	High-End Workstation	1-9
2-1	Bitmap Coordinates	2-2
2-2	Graphics Processor Internal Registers	2-5
2-3	Graphics Processor Status Register	2-5
2-4	Graphics Processor Instruction Pointer	2-6
2-5	Graphics Command Format	2-9
2-6	Sample Graphics Command Block (GCMB)	2-11
2-7	Opcode Register Used in Poll State	2-12
2-8	GP Registers Used in Exception Handling	2-12
2-9	Character Descriptor Block Format	2-19
2-10	Sample Character Descriptor Block	2-21
2-11	Byte Mode	2-22
2-12	Word Mode	2-22
2-13	Absolute Move	2-24
2-14	Draw Arc	2-26
2-15	Bit Block Transfer	2-28
2-16	Bit Block Transfer Between Two Bitmaps	2-30
2-17	Character Descriptor Block	2-33
2-18	Sample Character Descriptor Block	2-34
2-19	Word Mode Selector Words	2-35
2-20	Character Command in Word Mode	2-36
2-21	Byte Mode Selector Words	2-36
2-22	Draw Circle	2-38
2-23	Define Bitmap	2-40
2-24	Define Character Orientation	2-42
2-25	Character Descriptor Block	2-44

Figure	Title	Page
2-26	Sample Descriptor Block	2-44
2-27	Word Mode Character	2-46
2-28	Byte Mode Character	2-47
2-29	Define Clipping Rectangle	2-50
2-30	Incr_Point Descriptor Array	2-63
2-31	Array Values for Incr_Point	2-64
2-32	Draw Line	2-68
2-33	Draw Point	2-73
2-34	Draw Polygon	2-75
2-35	Polygon Vertices Array Information	2-75
2-36	Draw Polyline	2-76
2-37	Polyline Vertices Array	2-77
2-38	Draw Rectangle	2-79
2-39	Relative Move	2-80
2-40	Scan_Lines Horizontal Line Array	2-82
3-1	82786 and IBM PC Swapped Byte Bitmap Formats	3-2
3-2	IBM PC CGA Swapped Byte, 2 Bank Example	3-3
3-3	Swapped Byte, 4 Bank IBM PCJr Example	3-3
3-4	Screen Composition with Strips and Tiles	3-4
3-5	Irregular Shaped Window	3-5
3-6	Strip and Tile Descriptors	3-6
3-7	C Bit Can Indicate Final Strip Color	3-9
3-8	Zoom Tile Placement	3-14
3-9	Sample VRAM Design	3-16
3-10	Display Processor Internal Registers	3-19
3-11	DPStatus Register Bits	3-20
3-12	Display Control Block Registers	3-22
3-13	CRT Timing Signals	3-27
3-14	Video Data Pin Outputs	3-28
4-1	128-Byte Internal Register Block	4-2
4-2	BIU Register Map	4-3
4-3	Relocation Register	4-4
4-4	Internal Relocation Register Address	4-4
4-5	BIU Control Register	4-5
4-6	Refresh Scaler	4-6
4-7	DRAM/VRAM Control Register	4-8
4-8	Display Priority Register	4-9
4-9	Graphics Processor Priority Register	4-10
4-10	External Priority Register	4-10
4-11	82786 Synchronously Connected to an 80286	4-20
4-12	Asynchronous Slave 10 MHz 82786 Interface to 8 MHz 80186	4-24
5-1	Noninterleaved DRAM Arrays	5-2

Figure	Title	Page
5-2	DRAM Arrays with ×8 Devices	5-3
5-3	Correlation of Address with $\overline{RAS}$ and $\overline{CAS}$	5-4
5-4	External Multiplexers Provide 4 Bpp	5-6
5-5	Two Interleaved Banks of 256K × 1	5-8
5-6	Data Line Connections ×1	5-10
5-7	Data Line Connections ×4 and ×8	5-11
5-8	82786 Viewpoint of Graphics Memory Map	5-13
5-9	CPU Memory Map	5-14
5-10	Possible Memory Map for 80286/82786 System with Memory Mapped Internal Registers	5-15
5-11	Memory Mapping for 80286/82786 System with I/O Mapped Internal Registers	5-16
5-12	80186 System Accesses Only a Portion of Graphics Memory	5-17
5-13	Bank Switched Memory Allows 80286 to Access all of Graphics Memory	5-18
6-1	82786 Drives CRT with TTL-Level Inputs	6-3
6-2	Buffer Used to Drive TTL-Input CRT Interface	6-3
6-3	Analog CRT Interface Allows 256 Colors	6-4
6-4	Resistor Ladder	6-4
6-5	VDATA Color Assignments	6-5
6-6	Color Lookup Table	6-6
6-7	Color Lookup Table Circuit Generates 16 Video Bits from 8	6-7
6-8	Hybrid Color Lookup Table and DAC Simplifies Interface	6-8
6-9	Blank Demultiplexes Window Status Pins	6-8
6-10	Four Color Lookup Tables Selectable by Window Status Outputs	6-9
6-11	External Multiplexer Allows up to 50 MHz Video with 4 Bpp	6-10
6-12	Configuration for Video Data Rates up to 100 MHz	6-13
6-13	External ECL Shift Register Allows up to 200 MHz Video with 1 Bpp	6-14
6-14	Dual 82786s Generate 16 Bpp at 25 MHz	6-15
6-15	Two 82786s Create a 4-Bpp Display at 100 MHz	6-17
6-16	Character ROM and Bitmap Graphics on Same Screen	6-20
6-17	Very Large Character ROM and Bitmap Graphics on Same Screen	6-21
6-18	Dividing a Page into Strips Decreases Memory Requirements	6-25
6-19	Noninterlaced and Interlaced Displays	6-26
6-20	HSync and Blank Timing Parameters	6-28
6-21	VSync and Blank Timing Parameters	6-28
6-22	Display Processor Register 05H	6-30
B-1	82786 Pinout — Top View	B-1
B-2	Pinout 82786 — Bottom View	B-2

Tables

Table	Title	Page
2-1	Bits Per Pixel (bpp) and Pixels Per Word	2-3
2-2	Effective Address Variables	2-3
2-3	Graphics Processor Status Register Bits	2-6
2-4	Graphics Processor Control Registers	2-7
2-5	Context Registers	2-8
2-6	Drawing Command Attributes	2-14
2-7	Logical Operations	2-15
2-8	Logical Operations	2-52
2-9	Control Registers	2-56
2-10	Context Registers	2-57
2-11	Increment Point Codes	2-64
2-12	Incr_Point Command Array Values	2-65
3-1	Tile Descriptor Parameters	3-7
3-2	Possible CRT Displays with Standard DRAMs	3-11
3-3	Possible CRT Displays with VRAMs	3-11
3-4	HSync, VSync, and Blank Settings	3-11
3-5	Timing Signal Resolution Changes	3-13
3-6	BPP for Dot Rate Tradeoffs	3-13
3-7	DPStatus Register	3-20
3-8	Display Control Registers	3-23
4-1	RW1:0 Values	4-8
4-2	DC1:0 Values	4-8
4-3	HT2:0 Values	4-9
4-4	Default Priority Levels Following RESET	4-12
4-5	Suggested Priority Values	4-12
4-6	80286 Bus Master Performance	4-14
4-7	82786 Address Comparison	4-18
4-8	82786 Status Interface for the 80286	4-18
4-9	BHE# and MIO Pin Values	4-19
5-1	Memory Configurations	5-9
5-2	DRAM Single Cycle Times	5-11
5-3	DRAM Block Transfer Rates	5-11
6-1	Possible CRT Displays with Standard DRAMs	6-1
6-2	Possible CRT Displays with VRAMs	6-2
6-3	Valid StartBit and StopBit Values	6-11
6-4	HSync, VSync, and Blank Settings	6-22
6-5	Accelerated Video Mode Values	6-29
A-1	82786 Test Modes	A-1
B-1	Pin Descriptions	B-2

82786 Overview 1

CHAPTER 1
82786 GRAPHICS COPROCESSOR OVERVIEW

The 82786 is an intelligent graphics coprocessor that replaces subsystems and boards, which traditionally use discrete components and/or software for graphics functions. In a single 88-pin grid array or leaded carrier, the 82786 integrates a:

- Graphics Processor,
- Display Processor with a CRT controller, and a
- Bus interface unit with a DRAM/VRAM controller supporting 4 MB of memory, which can consist of both graphics and system memory.

The Graphics Processor (GP) and the Display Processor (DP) are independent processors on the 82786. The Bus Interface Unit (BIU) with its DRAM/VRAM controller arbitrates bus requests between the Graphics Processor (GP), Display Processor (DP), and the External CPU or Bus Master.

Figure 1-1 provides a functional overview of the 82786. Refer to Appendix B for top and bottom pin-out views and a description of each pin.

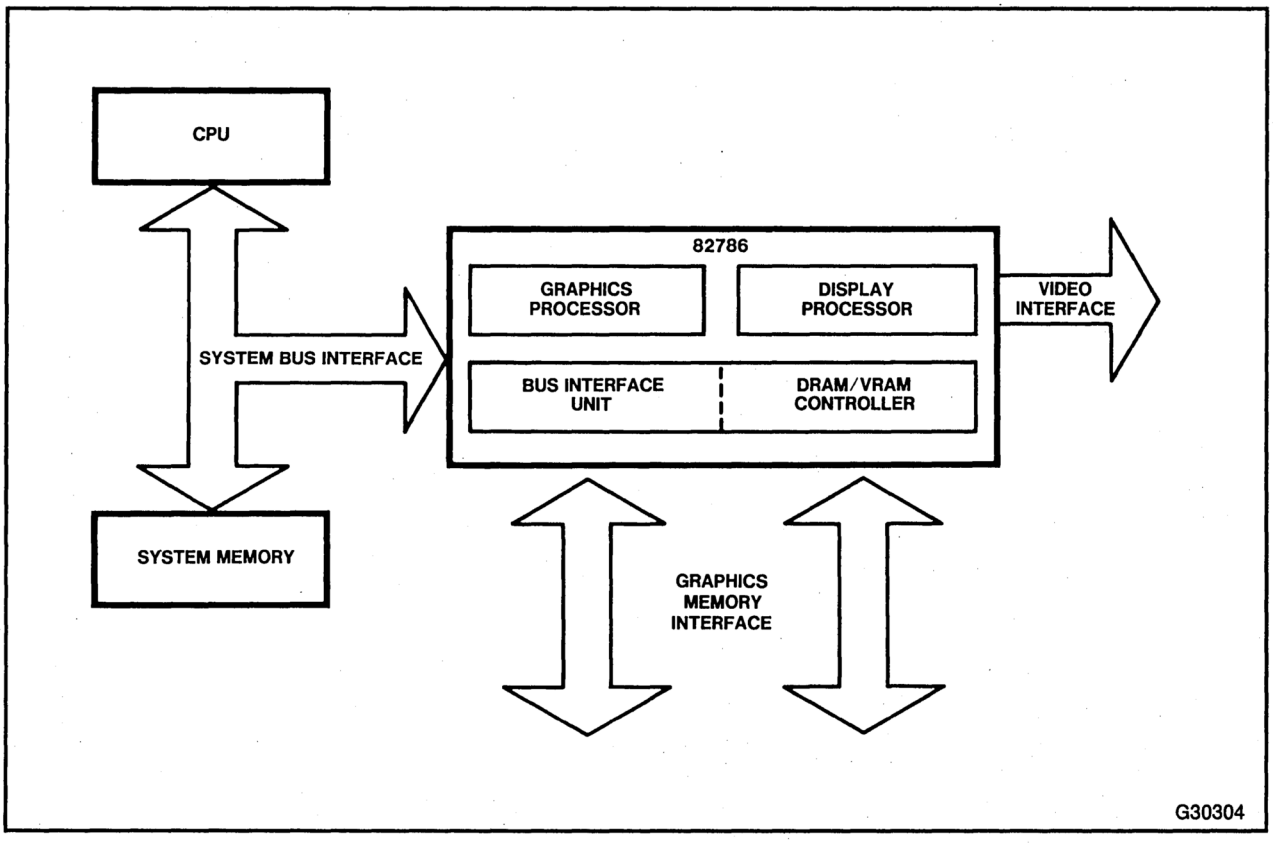


Figure 1-1. 82786 Functional Overview

The integrated design of the 82786 increases programming efficiency and overall performance while decreasing development and production time and costs of many microprocessor-based graphics applications such as personal computers, engineering workstations, terminals, and laser printers. Compatibility with Intel microprocessors, the many device independent standards, and IBM Personal Computer bitmap memory format (see Section 3.1.2 "IBM PC Bitmap Format Support") combined with support for international character sets, multitasking, and an 8- or 16-bit host makes programming the 82786 flexible and straightforward. The extensive features of the 82786 accommodate many designs. The list below contains some of the main 82786 features.

- Available in 88-pin grid array or leaded carrier
- Interface designed for device independent software
- Integrated drawing engine with a high-level computer graphics interface instruction set
- Supports multiple character sets (fonts) that can be used simultaneously for text display applications
- Hardware support for fast manipulation and display of multiple windows on the screen
- DRAM/VRAM controller supporting up to 4 MB of graphics memory, shift registers, and DMA channel
- Supports sequential access DRAMs and dual port video DRAMs (VRAMs)
- Fast bit-block transfers (bitblt) between system and graphics memory
- Supports up to 200 MHz CRTs or other video interface
- Up to 256 simultaneous colors per frame
- Programmable video timing
- Third-party software support
- Supports rapid pattern fill
- International character support
- Advanced CHMOS technology
- IBM Personal Computer bitmap formats

Support for high resolution displays using a 25 MHz pixel clock lets the 82786 display up to 256 colors simultaneously. Systems designed with multiple 82786s or a single 82786 with VRAMs can support virtually unlimited color and resolution.

1.0 REVISION INFORMATION

This revision (-003) of the *82786 Graphics Coprocessor User's Manual* reflects the features and capabilities of the "D" level of revision of the component, noted in this manual by the term "D-step."

1.1 ARCHITECTURE

The powerful yet flexible design of the 82786 requires minimal support circuitry for most applications, which reduces costs and board space requirements for many applications.

Also key to the 82786 is its memory structure. The 82786 can access either graphics memory directly supported by the integral DRAM/VRAM controller or external system memory that resides on the CPU bus. When the 82786 accesses system memory, it controls the bus and operates in Master Mode. The 82786 can also operate as a Slave with the CPU accessing the 82786 graphics memory and the Internal Registers. From the software standpoint, the 82786 accesses graphics and external system memory in the same manner. However, performance increases when the 82786 accesses its own graphics memory because the 82786 DRAM/VRAM controller accesses it directly without encountering contention with the CPU. Conversely, the CPU accesses its own system memory more quickly than graphics memory because it does not encounter contention from the Display Processor (DP) or Graphics Processor (GP).

Another feature of the 82786 is the bitmap organization. Replacing the traditional "bit plane" memory model, the 82786 utilizes sequential ordering (linear memory) and takes advantage of the fast sequential access modes of DRAMs or dual port video DRAMs (VRAMs) to gain performance. The 82786 supports a packed pixel bitmap organization for color in which all color bits for each pixel are stored in the same byte in memory. In the traditional bit plane model, each plane defines separate color information. For example, a 4-plane bitmap describes a bitmap with four colors as shown in Figure 1-2. Each byte of memory contains one bit of color information for each pixel in the 4-plane bitmap. In the 82786 packed pixel model, each byte stores data for two pixels. Section 2.1.3 "Pixels" describes the packed pixel bitmap in detail.

1.1.1 Graphics Processor (GP)

The 82786 Graphics Processor (GP) draws all geometric objects and characters and moves images within and between bitmaps. The GP creates and updates the bitmap, executes commands placed in memory by the host CPU, and updates the bitmap memory for the Display Processor (DP). The GP high-level commands provide high speed drawing of graphics objects and text. The GP performs all these functions independent of the DP. Refer to Chapter 2 for a detailed discussion of the GP and its functions.

1.1.2 Display Processor/CRT Controller (DP)

The Display Processor (DP) traverses bitmaps generated by the Graphics Processor (GP) or external CPU, organizes the data, and displays the bitmaps in the form of windows on the screen. The DP has a video shift register that can assemble several windows on the screen from different bitmaps in memory and zoom any of the windows in the horizontal and/or vertical directions. When the DP detects a window edge, it automatically switches to the next bitmap to display the subsequent window.

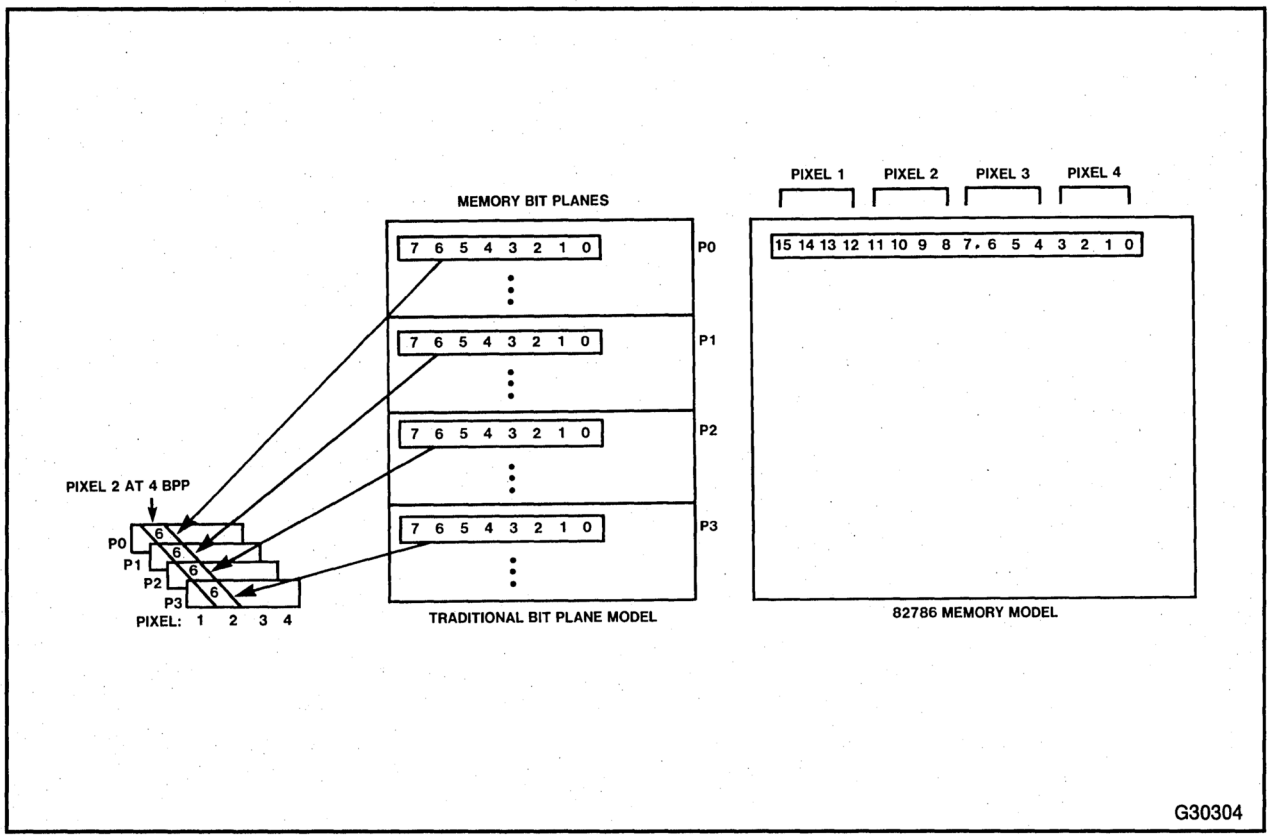


Figure 1-2. Sequential Ordering Replaces Traditional Bit Plane Model

Essentially, the DP operates as an address generator that accesses appropriate portions of memory-resident bitmaps. The data fetched from bitmaps is passed to the DP CRT controller, which displays the bitmap data on the screen. The DP CRT controller generates and synchronizes the Horizontal Synchronization (HSync), Vertical Synchronization (VSync), and Blank signals. The DP performs all these functions independent of the GP. Refer to Chapter 3 for a detailed discussion of the DP and its functions.

1.1.3 Display Processor (DP) Master and Slave Modes

The Display Processor operates as a Master or a Slave based on the Horizontal Synchronization (HSync) and Vertical Synchronization (VSync) signals, which are set with the S bit in the CRTMode Display Control Register (see Table 3-8 in Section 3.3.2 "Display Control Block Registers"). When the S bit is set to one, the DP is a slave with the HSync and VSync signals as inputs. If the S bit is 0, the DP operates as a Master with HSync and VSync as outputs. For details, refer to Section 3.2.1 "CRT Controller."

1.1.4 Bus Interface Unit (BIU)

The Bus Interface Unit (BIU) controls communication between the 82786, the external CPU, and graphics and external system memory when both are configured. A low-end system can use a single memory shared by the CPU and 82786 with the DRAM/VRAM controller

managing memory accesses as shown in Figure 1-4 in Section 1.2 "System Configurations." The BIU uses a DRAM/VRAM controller that supports dual port video DRAMs (VRAMs) and high speed burst access modes of page and fast page mode DRAMs. Both the GP and DP use the BIU to access bitmaps in memory. Refer to Chapter 4 for a detailed discussion of BIU concepts.

1.1.5 Memory Structure

The 82786 can address 4 MB of memory. Most systems divide memory in at least two segments: the 82786 graphics memory, which uses the DRAM/VRAM controller, and external system memory. Dividing memory can enhance the performance of graphics applications. The DRAM/VRAM controller allows faster access to graphics memory than external system memory because it does not encounter contention from the CPU. The CPU accesses system memory and executes programs simultaneously, while the 82786 accesses graphics memory and executes its commands. For sample system configurations, see Figures 1-5 and 1-6 in Section 1.2 "System Configurations."

However, when performance is not critical, the 82786 and CPU can share the same memory with the integral 82786 DRAM/VRAM controller managing memory accesses. With this configuration, target applications must be able to tolerate the decreased bandwidth of system memory. For a sample system configuration, see Figure 1-4 in Section 1.2 "System Configurations."

The 82786 assumes graphics memory starts at address 0H and ascends to the configured value, specified in the BIU DRAM/VRAM Control Register described in Section 4.2.4 "DRAM/VRAM Control Register." The 82786 can support a maximum of 4 MB of graphics memory, but the 82786 cannot access system memory if all its 4 MB addressing capacity is configured as graphics memory.

1.1.6 Memory Access and Arbitration

The BIU receives requests to access graphics memory from the Graphics Processor (GP), Display Processor (DP), and CPU. The BIU also receives memory refresh requests from the DRAM/VRAM controller. The BIU uses a priority system to arbitrate all requests. Memory refresh requests always have highest priority. Other requests have programmable priorities. A higher priority memory cycle can interrupt a lower one. For details, refer to Section 4.3 "Bus Cycle Arbitration."

1.1.7 Master and Slave Memory Access Interfaces

During memory access, the 82786 operates either as a Master or a Slave. The 82786 operates as a Master when it accesses external system memory. The 82786 acts as a Slave when the host CPU accesses graphics memory or any of the 82786 Internal Registers (see Section 1.1.8).

1.1.7.1 MASTER MODE INTERFACE

The 82786 operates as a Master whenever it accesses a memory address that is beyond the upper limit of configured graphics memory. Usually, this memory is external memory that the 82786 and CPU share. A high level on the Hold Request (HREQ) line indicates the 82786 is requesting the bus. The 82786 drives the external bus only after it receives a Hold Acknowledge (HLDA) from the External Bus Master. The HLDA is either externally synchronized (82786 synchronous mode) or internally synchronized (82786 asynchronous mode). The 82786 deactivates the HREQ when it no longer needs to access external memory or senses an inactive HLDA. The 82786 indicates that it controls the bus by a high level on the Master Enable (MEN) output. Details on the synchronous and asynchronous modes are discussed in Section 4.5 "System Bus Interface." For details on the Master Mode Interface, refer to Section 4.4.1.

1.1.7.2 SLAVE INTERFACE

As a Slave, the 82786 receives requests from the External Bus Master. For example, the host CPU accesses the 82786 graphics memory or its Internal Registers. The external CPU starts a slave access by asserting the Chip Select Low ($\overline{CS}$) input for a read/write to the 82786. When the 82786 is not Bus Master, the address lines (A21:0), Read Low ($\overline{RD}$), Write Low ($\overline{WR}$), Memory-I/O (MIO), and Byte High Enable Low ($\overline{BHE}$) lines are inputs. The 82786 constantly monitors the $\overline{RD}$, $\overline{WR}$, MIO, and $\overline{CS}$ lines to detect whether a CPU cycle occurred. The 82786 indicates the beginning of a Slave access by bringing Slave Enable (SEN) high and indicates the end of the access by bringing SEN low. The data bus transceiver can be enabled by SEN. For details on the Slave Interface, refer to Section 4.4.2.

1.1.8 Internal Registers

The 82786 has a 128-byte block of contiguous directly addressable Internal Registers, which is shown in Figure 1-3. The host CPU directly addresses this block of Internal Registers to communicate with the Graphics Processor (GP), Display Processor (DP), and Bus Interface Unit (BIU). The block can be either Memory or I/O mapped in the CPU address space. The base address and memory-I/O map (MIO) option are programmable through the BIU Internal Relocation Register, described in Section 4.2.1.

1.2 SYSTEM CONFIGURATIONS

The high performance and flexible features of the 82786 offer excellent solutions for a diverse range of applications and system configurations. Figures 1-4 through 1-6 illustrate just a few system designs.

In Figure 1-4, the 82786 combined with the 80186, shared system and graphics memory, and a monitor provide a low-end, low-priced, graphics system. In this system the CPU and the 82786 share memory and use the 82786 DRAM/VRAM controller for managing memory accesses. Target applications must tolerate the decreased bandwidth of system memory.

In Figure 1-5, the 82786 combined with an 80286, separate system and graphics memory, and a monitor provide an excellent multi-tasking office workstation.

For processing-intensive engineering environments, Figure 1-6 depicts a system with multiple 82786s in which each 82786 configures 4 MB of memory, an 80286 or 80386, and a monitor, which offers a powerful solution.

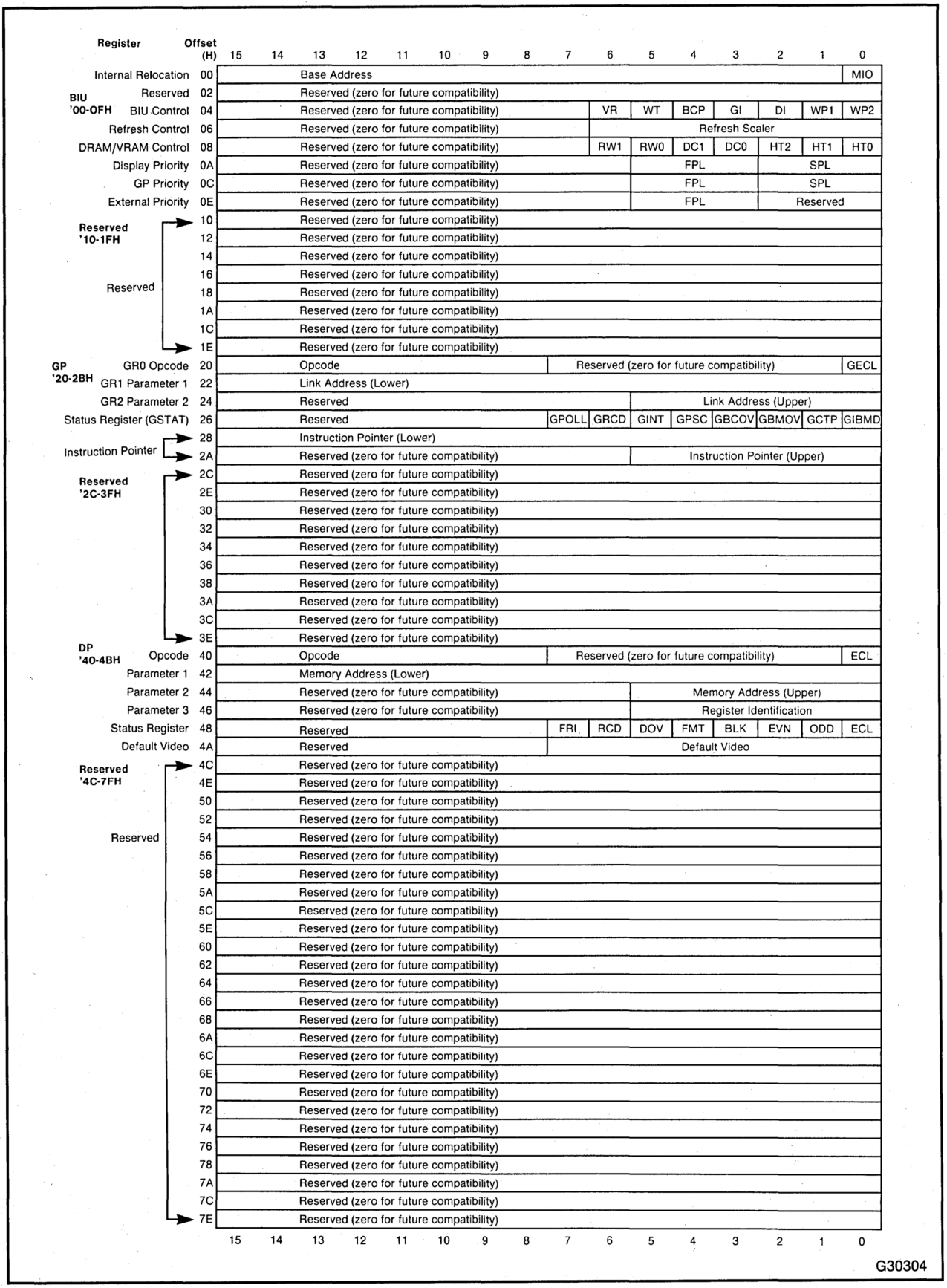


Figure 1-3. 82786 128-Byte Internal Register Block

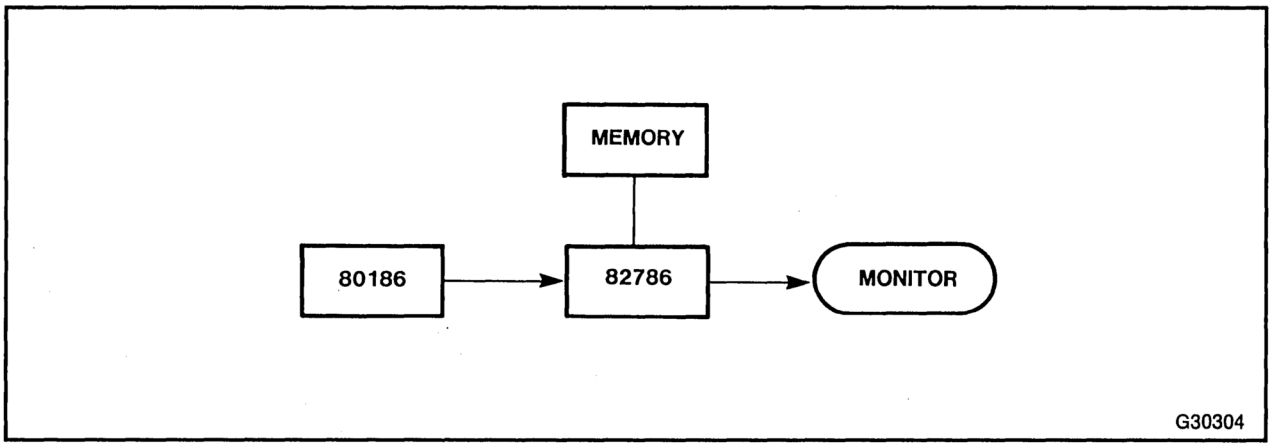


Figure 1-4. Low-End Low-Priced Personal Computer

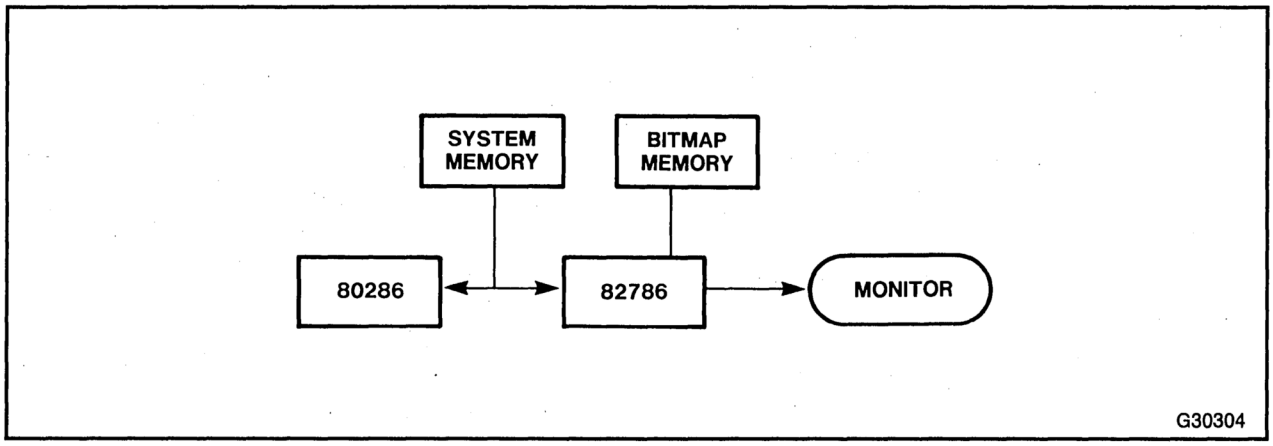


Figure 1-5. Multi-Tasking Office Workstation

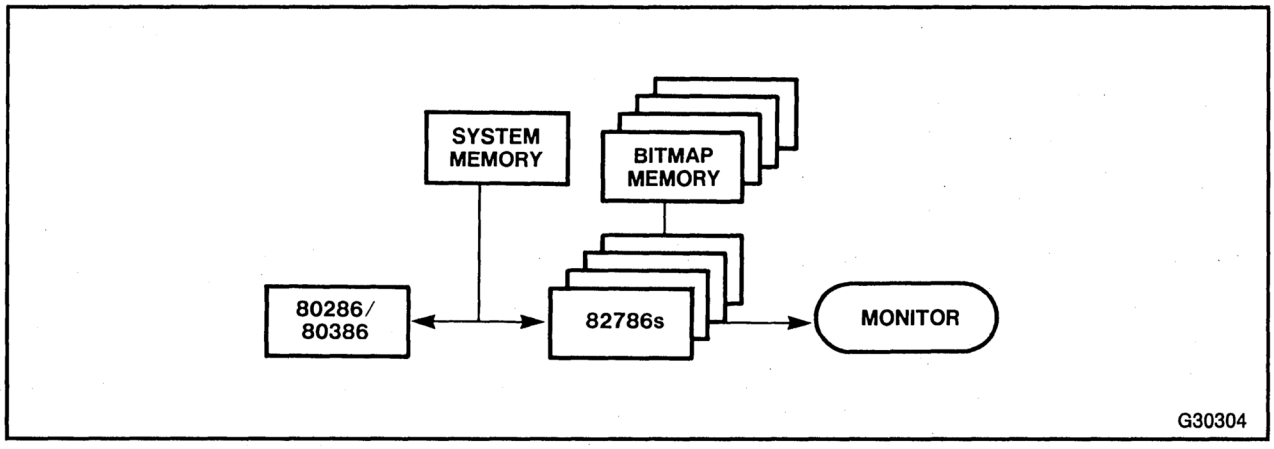


Figure 1-6. High-End Workstation

Graphics Processor Overview 2

CHAPTER 2
GRAPHICS PROCESSOR OVERVIEW

The Graphics Processor (GP) is an independent processor within the 82786 that creates and manages bitmaps in graphics or system memory. The GP creates and updates all graphics and text in each bitmap. The GP draws and moves all images in and between bitmaps. The list below outlines major functions provided by the GP.

- permits bitmaps to be any size (up to 32K × 32K pixels) and use 2, 4, 16, or 256 colors depending on the number of bits per pixel (bpp) which can be 1, 2, 4, or 8
- draws geometric shapes with attributes such as texture and color
- draws characters from user-defined fonts with attributes such as color, path, rotation, and proportional spacing
- combines one rectangular portion of a bitmap with another area, within the same bitmap or into another bitmap
- allows logical operations between source and destination (for example, logical Exclusive-OR or Complement of Source with Destination)
- clips drawings to a rectangular region
- supports picking, a mechanism used by user interfaces to select graphics menus (called icons) with a pointing device such as a mouse

2.1 GRAPHICS CONCEPTS

To use the 82786 effectively, an understanding of basic graphics concepts and how they relate to the 82786 is essential. The following sections discuss bitmaps, pixels, calculating the effective address of pixels, bitmap coordinates, and windows in the 82786 environment.

2.1.1 Bitmaps

The 82786 writes all graphics and text into bitmaps in memory. Each bitmap is a rectangular drawing area consisting of pixels that describe a graphic image or a character. The size of a bitmap varies based on the number of pixels and the number of bits associated with each pixel. A bitmap can be up to 32,768 pixels in each direction and contain 1, 2, 4, or 8 bits of color or gray scale information for each pixel. The amount of available memory determines the number of bitmaps that can exist.

A bitmap and the output display area do not necessarily correspond on a one-to-one basis. For example, a bitmap can be larger or smaller than the screen output area. Each bitmap has a specific number of bpp, whereas a screen can display several bitmaps; each with a different number of bpp. The Display Processor (DP) interprets the information residing within bitmaps and extracts a bitmap which can be positioned anywhere on the screen as a window. For details on DP operation, refer to Chapter 3.

Programming and performance efficiency can be improved by segmenting graphics and system memory, and using graphics memory and separate bitmaps for generating text and color graphics. The GP can access graphics memory faster than system memory because it does not encounter CPU contention. Two bitmaps can increase efficiency if, one bitmap contains text with one bit per pixel (bpp) and the other bitmap contains a multicolor graphic image with several bpp.

2.1.2 Bitmap Coordinates

Any location in a bitmap can be referenced through a set of (x,y) coordinates. As shown in Figure 2-1, the 82786 uses a set of coordinates whose Origin Address begins at coordinates (0,0), located in the top left corner. The x axis extends across the top and increases to the right. The y axis extends down the left side and increases from top to bottom.

2.1.3 Pixels

Most Graphics Processor (GP) commands manipulate a pixel or a group of pixels within a bitmap. A pixel is the smallest element that can be displayed on a CRT. A pixel can associate 1, 2, 4, or 8 bits with it. The number of bits per pixel (bpp) varies based on the attributes associated with the pixel. Although the attributes, hence the bpp, can vary, all pixels in the same bitmap must have the same number of bpp. Color and multiple gray scale displays require multiple bpp.

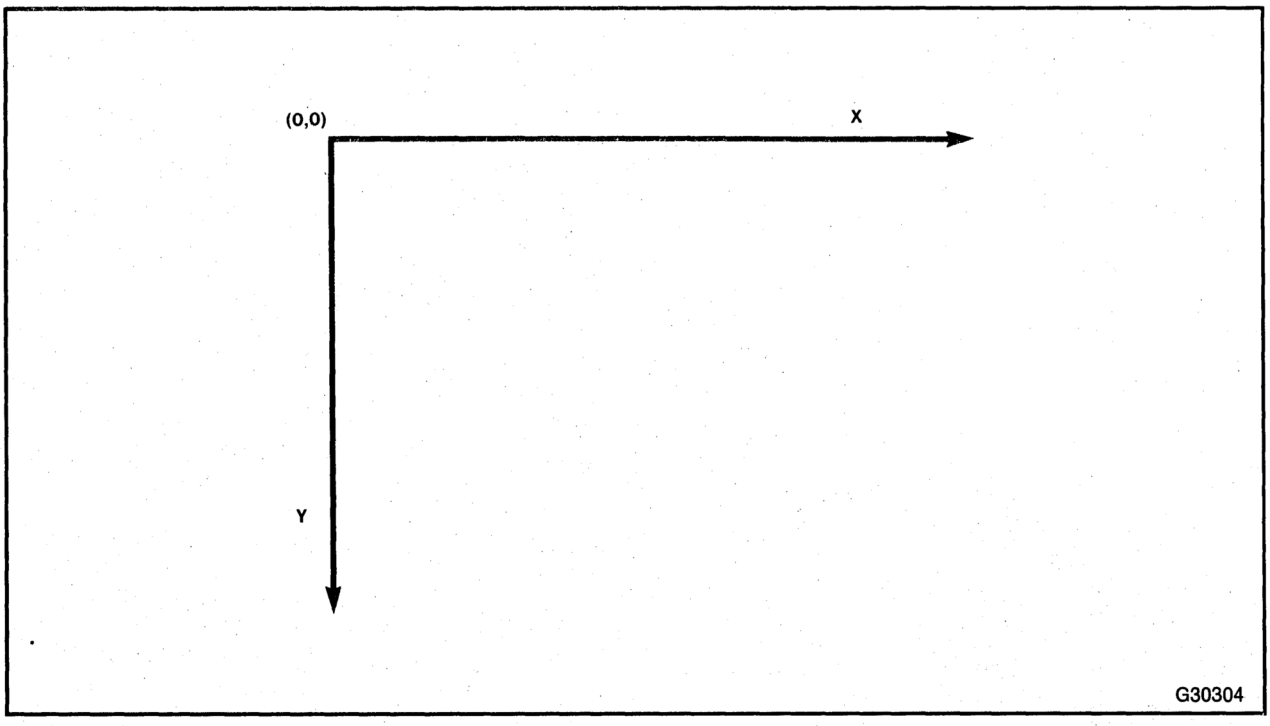


Figure 2-1. Bitmap Coordinates

All bitmaps are stored in memory in a sequential packed pixel manner. The 82786 stores pixels sequentially with as many pixels as possible in each 16-bit word. The number of pixels each word contains varies based on the number of bpp. For example, if the bitmap has 2 bpp, each word contains 8 successive pixels. Table 2-1 lists variations on the number of bpp and pixels per word.

Table 2-1. Bits Per Pixel (bpp) and Pixels Per Word

BPP	Pixels Per Word
1	16
2	8
4	4
8	2

The GP stores each bitmap as a series of bits, which the GP and Display Processor (DP) interpret as a series of lines consisting of sequentially ordered pixels. Lines are arranged from top to bottom and contain pixels ordered from left to right. The first pixel of every bitmap is the top left-most pixel. Each bitmap starts at the Origin Address, coordinates (0,0), which must be an even byte address because the GP and Display Processor (DP) address words only. The contents of each word varies based on the bpp. In the 2-bpp model, the first word of the bitmap holds the value of the first 8 pixels of the first line, (coordinates (0,0), (1,0), (2,0), ... (7,0)). The next word represents the next 8 pixels of the first line, and so on until the end of the line. The word containing the first 8 pixels of the second line (coordinates (0,1), (1,1) (2,1) ... (7,1)) follows the word containing the pixels for the end of the first line. The 82786 stores all bitmaps in this manner.

2.1.4 Calculating the Effective Address (EA)

To calculate the effective address of a pixel, refer to the formula below. Table 2-2 defines all variables.

EA (Effective Address) = Bitmap Origin Address + ((x*bpp) DIV 16) + y*N

bnum (starting bit position) = 15 − (x*bpp) MOD 16

Table 2-2. Effective Address Variables

Variable	Definition
bnum =	starting bit position of pixel (bnum = 0 is the least significant bit)
W =	Bitmap Width (in number of pixels)
H =	Bitmap Height (in number of pixels)
bpp =	Bits per Pixel
(x,y) =	Pixel Coordinates
N (Memory words per line) =	(W*bpp) DIV 16

Note: (W*bpp) must be an integral multiple of 16. The Bitmap Origin Address must point to a word (even byte) address.

2.1.5 Windows

Windows consist of one or more tiles, which are portions of bitmaps output by the Display Processor (DP). Horizontally, the 82786 supports up to 16 tiles per displayed line. Vertically, the 82786 can support the same number of tiles as scan lines. A tile must be a subset of a single bitmap, which can contain the entire bitmap, but not portions of two or more bitmaps.

2.2 GRAPHICS PROCESSOR (GP) REGISTERS

The Graphics Processor (GP) uses three types of registers:

- GP 82786 Internal Registers
- GP Control Registers
- Context Registers

All GP registers are 16 bits wide. Each register contains either data or an address. A data register can use all or just a portion of the 16 bits in the register. When the register value does not require all 16 bits, the value is right justified in the word and the upper unused bits are zeroed to ensure future compatibility. An address register requires two consecutive words to contain a 22-bit address. An address always begins on an even byte. Address registers do not explicitly store the least significant bit, which is assumed to be 0. The first word contains the 16 least significant bits. The second word contains the remaining six bits with the upper ten bits zeroed for future compatibility. The following sections discuss GP registers in detail.

2.2.1 Internal Graphics Processor (GP) Registers

The Graphics Processor (GP) has six Internal Registers on the 82786 that are either I/O or memory mapped. The external CPU uses these registers to initiate execution of graphics command lists and record status. Unlike other GP registers, which must be read and written to with Load and Dump Register commands, these GP registers are directly addressable. They are located at Internal Register Block offsets 20h through 2Ah from the base address (see Figure 1-3). The base address marks the beginning of the Internal Register Block and is contained in the BIU Relocation Register described in Section 4.2.1. Figure 2-2 displays these GP Internal Register Block registers which include GR0, the Opcode Register at offset 20h; GR1 and GR2, the lower and upper values of the starting address of the next Graphics Command Block (GCMB) at offsets 22h and 24h (see Section 2.3 "Command Execution and Format" for details); the GP Status Register (GSTAT) at offset 26 (see Section 2.2.1.1

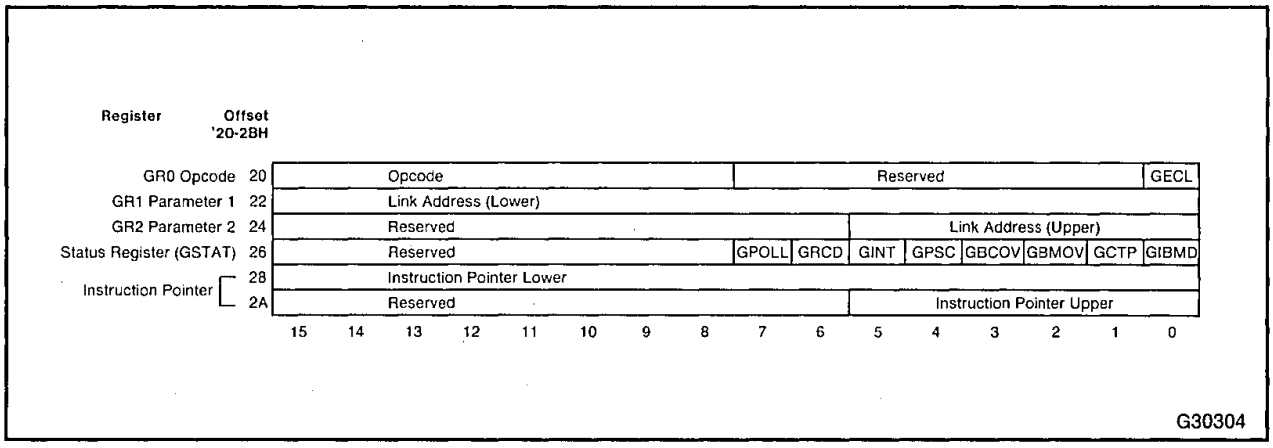


Figure 2-2. Graphics Processor Internal Registers

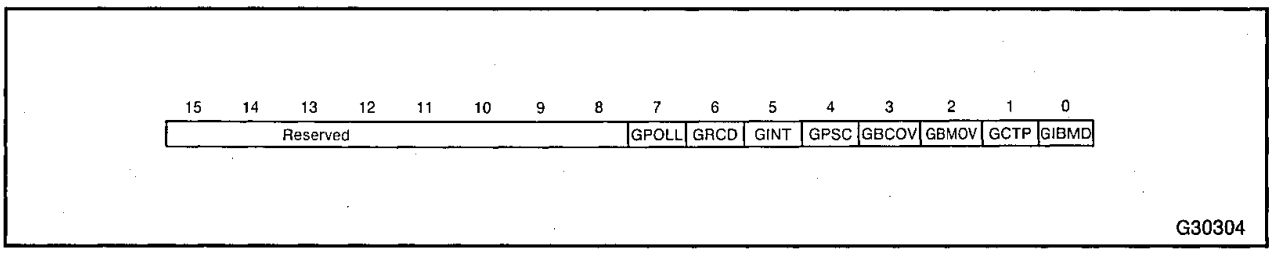


Figure 2-3. Graphics Processor Status Register

for details); and the Instruction Pointer (GCIP) at offsets 28h and 2Ah, (see Section 2.2.1.2 for details). These registers are word or byte addressable based on the setting of the BCP bit in the BIU Control Register described in Section 4.2.2 "Control Register."

2.2.1.1 GRAPHICS PROCESSOR STATUS REGISTER (GSTAT)

The Graphics Processor Status Register (GSTAT) contains the Status Byte, shown in Figure 2-3. Table 2-3 describes the GSTAT bits, which start at offset 26h. All the status bits, except GPOLL are cleared upon RESET. The GPOLL bit is set on RESET as discussed in Section 2.3.3 "RESET and Initialization."

2.2.1.2 GRAPHICS PROCESSOR INSTRUCTION POINTER

The Graphics Processor Instruction Pointer (GCIP) points to the current command in the Graphics Command Block (GCMB). The GCIP consists of a 22-bit value stored in two 82786 Internal Registers (GCIPL and GCIPH) at Internal Register Block offsets 28h and 2Ah from the base address as shown in Figure 2-4. The base address marks the beginning of the Internal Register Block and is contained in the BIU Relocation Register described in Section 4.2.1.

Table 2-3. Graphics Processor Status Register Bits

Bit	Function
GPOLL	Indicates if the Graphics Processor is in Poll State. See Section 2.3.2 "Poll State."
GRCD	Set if the Graphics Processor encounters an illegal opcode.
GINT	The INTR_GEN command, described in Section 2.10.21, sets this bit, which can cause an interrupt if the value in the GINT bit in the Interrupt Mask Register (GIMR) is zero. For details on the GIMR, refer to Section 2.3.4 "Exception Handling."
GPSC	Pick Successful Flag. Set or cleared while the Graphics Processor (GP) is in PICK Mode. Set if the Pick operation is successful on any command.
GBCOV	Bitmap Overflow. This flag is set when executing a Bitblt or Charblt command if any portion of the destination rectangle lies outside the clipping rectangle.
GBMOV	Bitmap Overflow Flag for geometric commands.This flag is set when drawing a pixel lying outside the clipping rectangle as a result of any geometric drawing command (Line, Circle, etc.).
GCTP	Character Trap. Indicates that a character had its Trap Bit set in the character string parameter of the Char command. This is useful in initiating error handling or special character handling routines such as those in which each character consists of multiple 16×16 pixel segments.
GIBMD	Illegal Bitmap Definition. Set if the Def_Bitmap command executes with illegal parameters. The illegal parameters are bits per pixel other than 1, 2, 4, 8, or Xmax defined to be greater than 32k−1. This bit should not be masked out in the Interrupt Mask (GIMR) to ensure the CPU is informed of this exception as soon as it occurs.

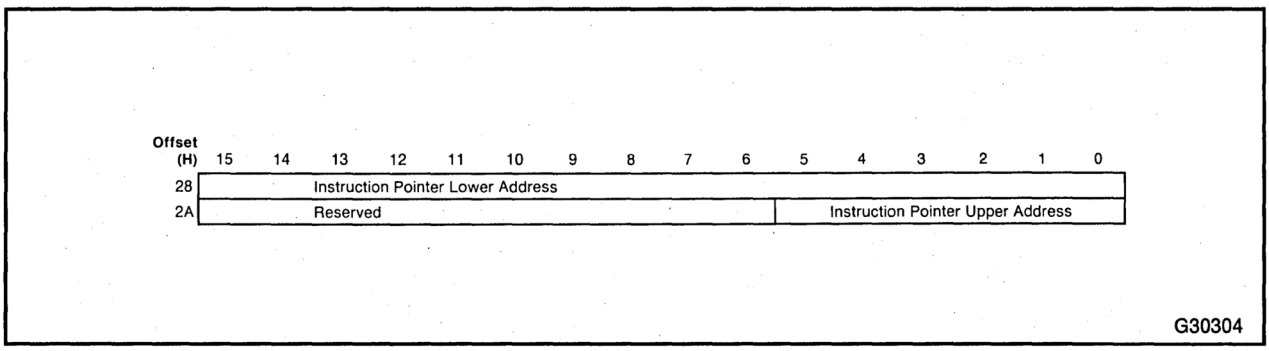


Figure 2-4. Graphics Processor Instruction Pointer

2.2.2 Graphics Processor (GP) Control Registers

The Graphics Processor (GP) has four Control Registers in addition to the Internal Register Block registers:

Register	Mnemonic
Poll Mask	GPOEM
Interrupt Mask	GIMR
Stack Pointer	GSP
Character Count	GCNT

Table 2-4 lists the function and ID of each register.

Table 2-4. Graphics Processor Control Registers

Register	ID	Number of Bits	Function
GPOEM	0003	6	Poll Mask Six right justified bits in a 16-bit register corresponding to GSTAT bits: GINT, GPSC, GBCOV, GBMOV, GCTP, and GIBMD. If any GPOEM bits corresponding to set GSTAT bits have a value of zero, the GP enters Poll State.
GIMR	0004	8	Interrupt Mask Eight right justified bits in a 16-bit register corresponding to GSTAT bits: GPOLL, GRCD, GINT, GPSC, GBCOV, GBMOV, GCTP, and GIBMD. The GP generates an interrupt if any GIMR bits corresponding to set GSTAT bits have a value of 0.
GSP	010C	21	Stack Pointer The 21-bit address contained in two consecutive words points to the current top of stack. The stack grows toward lower addresses.
GCNT	0015	16	Contains the character count while drawing characters in the bitmap.

The registers in Table 2-4 can be read from or written to with the Dump_Reg (Section 2.10.15) and Load_Reg (Section 2.10.24) commands. Each register is identified by a 9-bit Register ID, which is right justified in the lower nine bits of a 16-bit word in which the upper bits are zeroed.

GPOEM, GIMR, and GCNT are data registers, which always store their values right justified in a 16-bit word of memory in which the upper unused bits are zeroed. The Stack Pointer (GSP) requires two consecutive words of memory to hold its 22-bit address. The first word contains the lower 16 bits; the second word contains the six most significant bits in bits 0 through 5 with the remaining upper bits zeroed. The stack grows down toward lower addresses.

2.2.3 Graphics Processor (GP) Context Registers

The Graphics Processor (GP) also uses context registers, which should not be accessed normally, but must be saved and restored if the GP is to be shared by multiple processes. Any other access to these registers should be avoided. Table 2-5 lists these registers, their IDs, and the number of bits each uses. A register using 16-bits or less stores its value in a 16-bit word. If the register does not require the entire 16 bits, the value is right justified in the word and the upper bits are zeroed, unless noted otherwise. A register with a value larger than 16 bits uses two consecutive 16-bit words of memory. The first word contains the lower 16 bits and the second word contains the upper bits with any remaining bits zeroed.

Table 2-5. Context Registers

Name	ID	Bits	Function
GCOMM	0002	(16)	Command
GPOEM	0003	(6)	Poll Mask
GIMR	0004	(6)	Interrupt Mask
GCHOR	0007	(2,2)	Character Orientation and Path*
GCHA	010B	(21)	Character Font Base Address~
GSP	010C	(21)	Stack Pointer~
GCA	010D	(21)	Memory Address of Current Position (x,y)~
GBORG	010F	(21)	Bitmap Origin Address~
GCX	0010	(16)	Current X Position
GCY	0011	(16)	Current Y Position
GPAT	0012	(16)	Line Pattern
GSPAC	0013	(16)	Spacing between characters and Bitblts
GCNT	0014	(16)	Character Count**
GN	0016	(16)	Number of 16-bit words spanning width of bitmap
GVERS	0017	(16)	Version Number*** (D Step Value = 5)
TEMP	0019	(16)	Temporary Storage
GXMAX	0090	(16)	Maximum X for Clipping Rectangle
GYMAX	0091	(16)	Maximum Y for Clipping Rectangle
GXMIN	0094	(16)	Minimum X for Clipping Rectangle ^
GYMIN	0095	(16)	Minimum Y for Clipping Rectangle ^
GMASK	0099	(16)	Pixel Mask
GBGC	009B	(16)	Background Color
GFGC	009C	(16)	Foreground Color
GFCODE	009E	(4)	Function Code ^ ^
GCIP	01AC	(21)	Current Instruction Pointer~
GBPP(RO)	009F	(4)	Used with Dump Register command to get Current Bits per Pixel Value ^ ^ ^
GBPP(WO)	0008	(4)	Used with Load Register command to write Current Bits per Pixel Value ^ ^ ^

~ 21-bit registers use 2 consecutive words.

* These bits are right justified in each byte of the word in which they are stored. Two bits are stored in bits 1 and 0 and two bits are stored in bits 8 and 9; the remaining upper bits in each byte are zeroed.

** GCNT ID reassigned from 0015 to 0014 in D-Step.

*** In D-Step, valid after RESET and prior to drawing or drawing control commands.

^ Correction to previous GXMIN ID 0096 and GYMIN 0097 assignments.

^ ^ GFCODE ID reassigned from 001C to 009E in D-Step.

^ ^ ^ New D-Step Bpp Registers.

NOTE: The following information is not saved by saving the state of these registers:

1) Type of character font, word or byte.
2) Whether or not you are in pick mode.
3) Transparent or opaque drawing.

2.3 COMMAND EXECUTION AND FORMAT

The Graphics Processor (GP) fetches its commands from a Graphics Command Block (GCMB), which is a linked command list in memory. Although the GCMB can exist in system memory, the GCMB should be placed in graphics memory, controlled by the integral DRAM/VRAM controller, to take advantage of the faster command access time. The GP

82786 Internal Registers GR0, GR1, and GR2, shown in Figure 2-2 in Section 2.2.1, are loaded with the address of the GCMB and a Link command to initiate execution. The following sections discuss the GCMB format, Poll State, exception handling, and RESET.

2.3.1 Graphics Processor Command Block (GCMB) Format

The Graphics Processor (GP) processes Graphics Command Blocks (GCMBs), consisting of memory-resident commands. The GP runs only when an application needs to change the bitmap contents or support a special function such as picking (see Section 2.6.5). Figure 2-5 shows the format of a graphics command.

Each command in a Graphic Command Block consists of an opcode, a Graphics End of Command List (GECL) bit, and a list of parameters required by the command. The opcode is 8 bits wide and resides in the high byte. Bits 7 through 1 must be all zeroes to ensure future compatibility. Bit 0 is the GECL bit, which tells the GP to start or stop processing a command.

The 82786 uses a 22-bit address, but reserves 32 bits for all addresses. The 10 most significant unused bits must be zeroes. All GCMBs and commands they contain must lie at even byte addresses because the GP and Display Processor (DP) address words only.

The opcode is in the high byte. The GECL bit, bit 0, is the least significant bit of the low byte. A varying number of parameters follow in consecutive words. The GP tests the GECL of each command. If the GECL is set to 0, the GP processes the command. If the GECL is set to 1, the GP does not process the command and enters Poll State (see Section 2.3.2). Poll State halts the GP until:

- the upper and lower memory values denoting a link address are loaded into registers GR1 and GR2,
- a Link command is loaded in the Opcode Register GR0, and
- the GECL bit is zeroed.

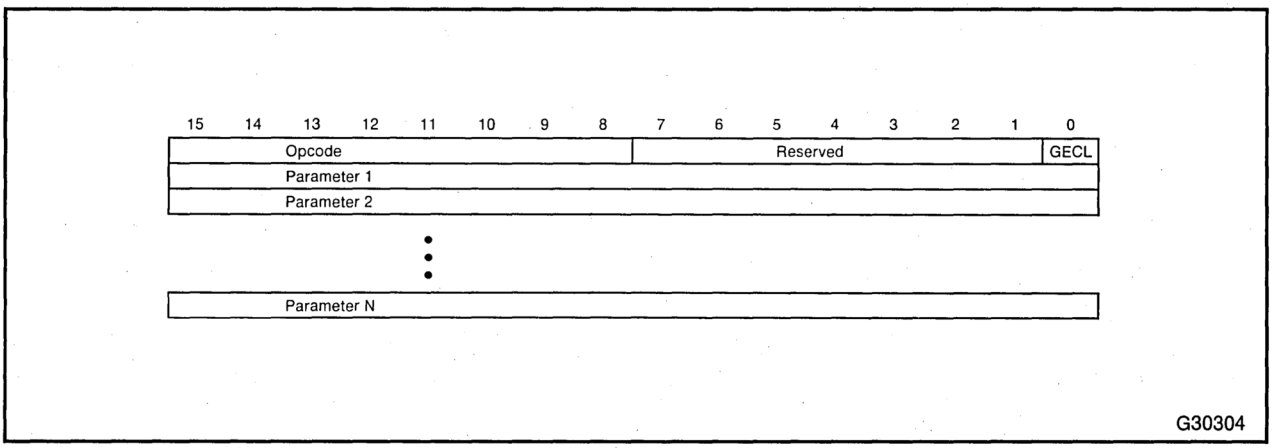


Figure 2-5. Graphics Command Format

The Link command in the Opcode Register must reset the GECL bit to 0. Then, the GP executes a new GCMB, which starts at the address specified in registers GR1 and GR2. Figure 2-6 shows a sample GCMB which illustrates how the Control Registers provide a link address to access memory-resident commands. The GCMB contains several Link commands that execute commands in other memory segments. It also contains a Call command that calls a graphics subroutine. A Nop command combined with the Halt command stops command execution.

2.3.2 Poll State

Poll State is the Graphics Processor (GP) default state after a RESET. The GP also can enter Poll State following command execution, an exception, a software abort signal, or if the GRCD bit in the Status Register (GSTAT) is set (see Section 2.2.1.1 "Graphics Processor Status Register (GSTAT)" for details). While in Poll State, the GP does not execute commands and is considered to be idle. However, the GP continuously monitors the GECL bit in the Opcode Register, shown in Figure 2-7.

A valid command in the Opcode Register, which zeroes the GECL bit, starts the GP. After RESET, the first command placed in the Opcode Register must always be a Link command directing the GP to a GCMB in memory. If the GECL bit is a one, the GP does not execute the command specified by the opcode, instead, the GP enters Poll State. For example, the Halt command (Section 2.10.19) used with the Nop command (2.10.25) stops GP command execution.

Normally, the Graphics Processor (GP) returns to Poll State following command execution and the GECL bit being set to 1. The GP also can be forced to enter Poll State by receiving a software abort when either of the following events occur:

- Writing to the Status Register (GSTAT), described in Section 2.2.1.1.
- Writing to the Instruction Pointer (GCIP), described in Section 2.2.1.2.

When the GP receives the software abort, it enters Poll state after executing the current command.

2.3.3 RESET and Initialization

Upon RESET, the Graphics Processor (GP) enters a well defined state with the following events occurring:

1. Command execution halts and the GP enters Poll State.
2. The GECL bit in Opcode Register (GR0) is set to 1 indicating the end of the command list.
3. All status bits except GPOLL are cleared; GPOLL is set.
4. Interrupt Mask Register (GIMR) is set to all ones to disable it.
5. Poll On Exception Mask Register (GPOEM) is set to all ones to disable it.
6. GP exits Pick Mode.

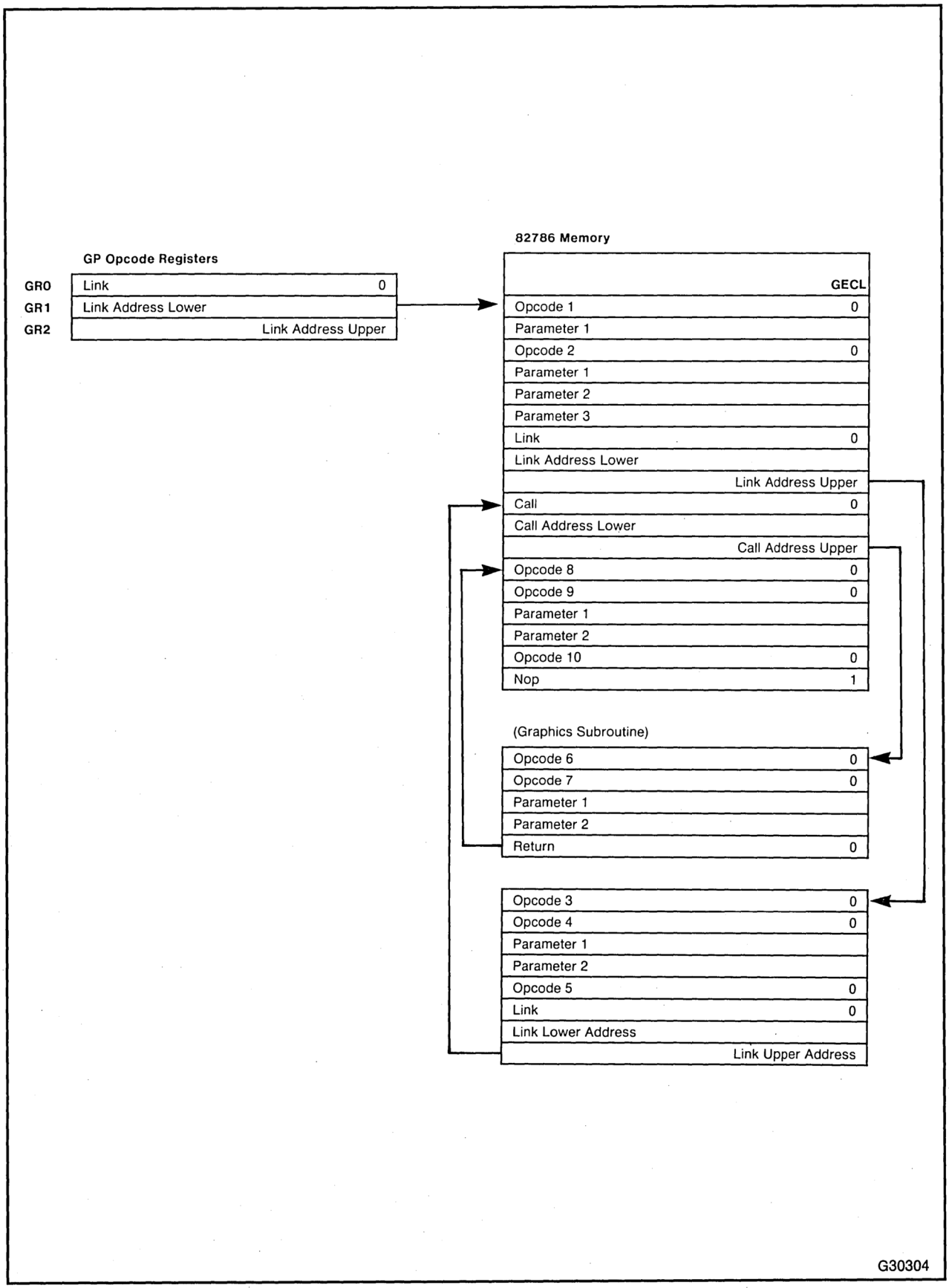


Figure 2-6. Sample Graphics Command Block (GCMB)

2.3.4 Exception Handling

The Graphics Processor (GP) either generates an interrupt or enters Poll State based on the status of the bits in the Interrupt Mask Register (GIMR), the GRCD bit in the Status Register (GSTAT), and the Poll On Exception Mask Register (GPOEM).

The Interrupt Mask GIMR has 8 bits corresponding to the GSTAT bits: GPOLL, GRCD, GINT, GPSC, GBCOV, GBMOV, GCTP, and GIBMD (see Figure 2-8 and Table 2-3 in Section 2.2.1.1 "Graphics Processor Status Register (GSTAT)." The GP generates an interrupt when any bit in the GIMR with a value of zero corresponds to a set bit in GSTAT. Another interrupt is not generated if the previous interrupt has not been acknowledged. Reading GSTAT and the BIU Control Register serves as an Interrupt Acknowledge to the GP; the interrupt and status bits not masked out by the Interrupt Mask are cleared. If the GPOLL bit causes the interrupt, it is not cleared on an Interrupt Acknowledge. However, this does not generate repeated interrupts.

In addition to a software abort signal or a Halt command, the GP enters Poll state when:

- The Status Register (GSTAT) Illegal Opcode GRCD bit is set.
- Any of the six GPOEM bits corresponding to set GSTAT bits GINT, GPSC, GBCOV, GBMOV, GCTP, or GIBMD have a value of zero. Refer to Figure 2-8 and Table 2-3 in Section 2.2.1.1 "Graphics Processor Status Register (GSTAT)."

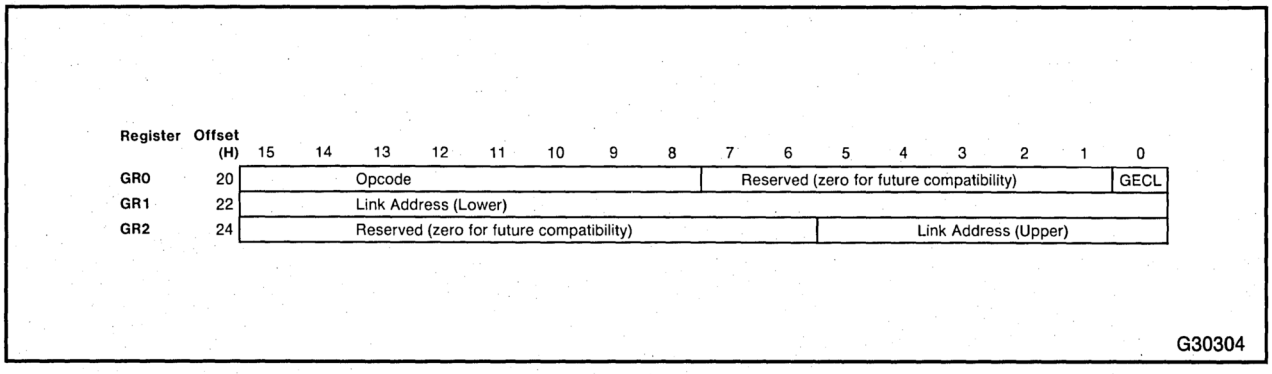


Figure 2-7. Opcode Register Used in Poll State

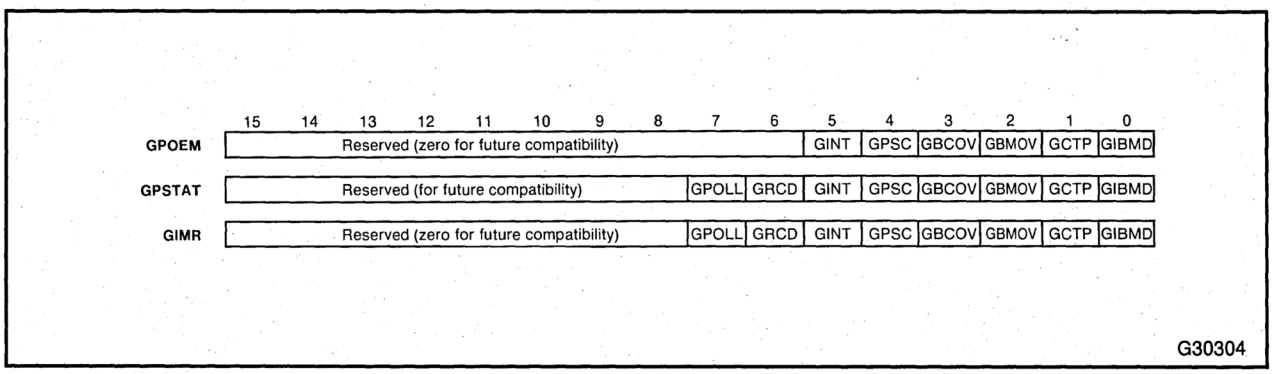


Figure 2-8. GP Registers Used in Exception Handling

Any time the GP enters POLL State, the GECL bit in the Opcode Register, GR0, automatically is set to one. When the GP is in POLL State, it can be restarted by writing a valid opcode such as 02H, the Link command, into register GR0 and writing a 0 into the GECL bit, which directs the GP to the graphics command block (GCMB) in memory. Clearing the GECL bit in this manner, also clears the status bits that caused the POLL State.

Interrupt and Poll are two independent functions. The GP can issue an interrupt and not POLL, or issue an interrupt and POLL, or not issue an interrupt and POLL, or do none of these functions.

2.4 GRAPHICS PROCESSOR (GP) COMMANDS

Graphics Processor (GP) commands can be divided into five classes:

1. Nondrawing
2. Drawing Control
3. Geometric
4. Bit Block Transfer (BitBlt)
5. Character Block Transfer (CharBlt)

The following sections list commands and discuss concepts.

2.5 NONDRAWING COMMANDS

Nondrawing commands control command fetching, and loading and dumping of the GP Control and Context Registers. These commands are:

Command	Function
Link	Link To Next Command (Unconditional Jump)
Call	Call Subroutine
Return	Return from Subroutine
Intr_Gen	Generate Interrupt
Dump_Reg	Dump Register
Load_Reg	Load Register

For details on any of these commands, refer to the specific command in the Command Description Section 2.10.

2.6 DRAWING CONTROL COMMANDS

Drawing control commands define the current bitmap and its attributes, and set the Graphics Current Position Pointer (GCPP).

Command	Function
Def_BitMap	Define Bitmap
Def_Clip_Rect	Define Clipping Rectangle
Def_Colors	Define Colors
Def_Texture	Define Texture
Def_Logical_Op	Define Logical Operation
Def_Char_Set	Define Character Set
Def_Char_Orient	Define Character Orientation
Def_Char_Space	Define Intercharacter Spacing
Abs_Mov	Absolute Move GCPP
Rel_Mov	Relative Move GCPP
Enter_Pick	Enter Pick Mode
Exit_Pick	Exit Pick Mode

For details on any of these commands, refer to the specific command in the Command Description Section 2.10.

2.6.1 Attributes Associated with Drawing Commands

A drawing operation modifies pixels within a bitmap during the execution of graphics commands. All drawing that the Graphics Processor (GP) performs (including lines, arcs, characters, and bitblts) are subject to several attributes which must be defined before executing any drawing command. The attributes and the commands that define them are listed in Table 2-6. The following sections discuss each of these attributes in detail.

Table 2-6. Drawing Command Attributes

Attribute	Initialized By
Color Bit Mask	Def_Logical_Op
Logical Operation	Def_Logical_Op
Clipping Rectangle	Def_Clip_Rect
Foreground/Background Color	Def_Colors
Transparent or Opaque Mode	Def_Texture or T/O bit in Charblt command*
Pattern Mask (texture)	Def_Texture
Pick Move	Enter_Pick

***Note:** Foreground and background color, Transparent and Opaque Modes, and Pattern Mask are not applicable to bitblt and character block transfer (charblt) commands. However, charblt commands use the T/O bit in the Char command to specify Transparent and Opaque Modes as well as the active foreground and background colors. For details on the T/O bit in the Char command, refer to 2.10.6 "CHAR – Draw Character String."

2.6.2 Color Bit Mask

The color bit mask defined in the Define Logical Operation command (Def_Logical_Op in Section 2.10.14) restricts the effect of graphics primitives on the bit plane to update a subset of the bits per pixel. With the color mask, one set of drawings can exist in one or more colors and other text or graphics information can reside in different color bits of the same bitmap.

2.6.3 Logical Operations

Logical operations can combine existing pixel information in a bitmap with new pixel information generated as a result of a drawing operation, such as displaying only the overlapping regions of two shapes. This operation logically combines the contents of separate bitmap locations to produce new bitmap patterns. The logical operation attribute, defined with the Def_Logical_Op command described in Section 2.10.14, applies to all subsequent pixel update operations (line, arc, charblt, bitblt, etc.) Sixteen binary functions are permitted between the source and destination as shown in Table 2-7.

The Fcode is the truth table of the associated two variable functions with the truth table of the FCode of function ABCD diagramed as follows:

Destination	Source 0	Source 1
0	A	B
1	C	D

Table 2-7. Logical Operations

Function	Fcode (Hex)
0	0000
source AND destination	0001
CMP (source) AND destination	0002
Destination	0003
source AND CMP (destination)	0004
source	0005
source XOR destination	0006
source OR destination	0007
CMP (source) AND CMP (destination)	0008
CMP (source) XOR destination	0009
CMP (source)	000A
CMP (source) OR destination	000B
CMP (destination)	000C
source OR CMP (destination)	000D
CMP (source) OR CMP (destination)	000E
1	000F

The following example shows the Complement of Source (CMP) function (FCode = 000Ah), which replaces the destination with the complement of the source. The truth table shows that if the source equals zero, the destination value does not care, so the result always will be one. It also shows that if the source is one, the destination value does not care, so the result always will be zero.

Destination	Source 0	Source 1
0	1	0
1	1	0

2.6.4 Clipping Rectangle

The clipping rectangle limits the effects of drawing operations to a subset of the bitmap. The Define Clipping Rectangle command (Def_Clip_Rect in Section 2.10.12) defines the clipping rectangle as any rectangle within the bitmap. The default clipping rectangle is the entire bitmap. The clipping rectangle prevents drawing outside the defined rectangular region. The clipping rectangle must be defined after a Def_Bitmap command (see Section 2.10.8).

Pixels are not drawn beyond the clipping rectangle. For figures that are partially inside and partially outside the clipping rectangle, only the part that is inside the clipping rectangle is updated in the bitmap. For block transfer and character drawing operations, if any part of the destination area lies outside the clipping rectangle, that outside part is not drawn. To ensure predictable results, the following restrictions apply to the clipping rectangle:

- For lines, circles, polygons, polylines, bitblts, and character block transfers (charblts) each pixel lying on the figure (both visible and the invisible parts) must not have its X or Y coordinate outside the 32K range.
- For circular arcs, the above restriction applies to the circle of which the arc is a part.

Refer to Section 2.10.6 for more information on character clipping.

2.6.5 Pick Mode

Pick Mode uses the clipping rectangle, which can be software controlled, to support selection of objects on the display by a pointing device. Drawing, bitblt, and character block transfer (charblt) commands are executed, but pixels are not updated in memory. Instead, if any pixels generated by the drawing command lie within the clipping rectangle, the Pick Successful Flag (GPSC) in the Status Register (GSTAT) is set. This allows the clipping rectangle to be set to correspond with the location of a graphics pointing device (for example, a cursor or pointer) and the Graphics Command Block (GCMB) can be reprocessed to find which drawing command corresponds to the selected area. Refer to Section 2.2.1.1 for details on GSTAT.

To put the Graphics Processor (GP) in Pick Mode, define the clipping rectangle with the Def_Clip_Rect command (see Section 2.10.12) and then execute the Enter_Pick command (see Section 2.10.17).

Pick mode is not supported for Circle and Arc commands.

2.6.6 Foreground and Background Colors

The Define Colors command (Def_Colors), described in Section 2.10.13, sets the foreground and background colors, the two colors drawn by all drawing operations (if both are needed). This command is not applicable for bitblt. Character block transfer (charblt) commands use the T/O bit in the Char command to select the active foreground and background colors based on those set by the Def_Colors command. For details on the Char command, refer to Section 2.10.6 "CHAR – Draw Character String."

2.6.7 Transparent or Opaque Mode

The Define Texture command (Def_Texture), described in Section 2.10.15, determines whether Transparent or Opaque Mode is used for most drawing and geometric commands. Transparent Mode draws only the foreground color into the bitmap (for dotted lines or characters) and does not change the pixels between dots or characters. Opaque Mode draws the foreground color and fills in the background color between dots or characters. Neither of these modes is applicable for bitblt or character block transfer (charblt) commands. Charblt commands use the T/O bit in the Char command (see Section 2.10.6) to define Transparent or Opaque Modes.

2.6.8 Pattern Mask

The pattern defined in the mask initiates a logical operation with drawing commands, which permits dotted and dashed lines, arcs, and other shapes. This attribute is not applicable for bitblt and character block transfer (charblt) commands. The Def_Texture command (see Section 2.10.15) sets the transparent/opaque attribute for drawing operations other than those that the Char command (see Section 2.10.6) defines.

2.7 GEOMETRIC COMMANDS

Geometric commands draw points, lines, and arcs in a variety of ways as the following list illustrates.

Command	Function
Point	Draw Point
Incr_Point	Draw Incremental Points
Circle	Draw Circle
Line	Draw Line
Rect	Draw Rectangle
Polyline	Draw Polyline
Polygon	Draw Polygon
Arc	Draw Arc
Scan_Lines	Draw Series of Horizontal Lines

For details on any of these commands, refer to the specific command in the Command Description Section 2.10.

2.8 BIT BLOCK TRANSFER (Bitblt) COMMANDS

Bit Block Transfer (bitblt) commands combine rectangular images from one piece of bitmap memory to another.

Command	Function
Bit_Blt	Bit Block Transfer within a bitmap
Bit_Blt_M	Bit Block Transfer between bitmaps

For characters larger than native (16×16 pixels), if the font is arranged as a bitmap, Bit_Blt_M can be used to write arbitrary sized characters. For details on either of these commands, refer to the specific command in the Command Description Section 2.10.

2.9 CHARACTER COMMAND

The Character command (Char) allows an application using character codes such as ASCII to draw character fonts stored in memory into a bitmap. The character fonts are stored in pixel form. For details on the Char command, refer to Section 2.10.6 in the Command Description Section. The following sections discuss font support and character storage.

2.9.1 Character Font Support

The Graphics Processor (GP) supports an unlimited number of character fonts, which can reside anywhere in the 4 MB address space. The Def_Char_Set command, described in Section 2.10.10, defines whether the character string is a string of bytes or a string of words. The type of font used generally determines the mode that you should select. For fonts coded

in ASCII, Byte Mode can be used because ASCII character codes are 8 bits. However, for Kanji, Word Mode is necessary because more than the 256 character limit supported in Byte Mode is needed.

The Def_Char_Set command sets the active font type and the base memory address of the font. The Def_Char_Space command (see Section 2.10.11) can be used for negative inter-character spacing to permit kerning for italic fonts and special effects.

2.9.2 Character Storage

Each character in a character font has an independently programmable size of up to 16×16 pixels, which allows individual characters to be different sizes for proportional spacing. Information about each character resides in memory and is provided by the user. Starting from an even byte address, the character information is stored in consecutive words of memory forming a character block containing n+1 words of memory, where n is the pixel height of the character. Each Character Descriptor Block has the format shown in Figure 2-9.

Each Character Descriptor Block must start at an even byte address. The dot patterns for each row must reside in consecutive words in memory. The first word defines the height and width of the character (see Figure 2-9) and whether the character is an overstrike or exceeds the 16×16 pixel limit in either direction. If the dot pattern for the width of the character is less than 16 pixels, the dot pattern for the row must be right justified.

Each character is identified by specifying the address of the starting block. This address is an offset from the character table base address, which is programmed before using the Def_Character_Set command.

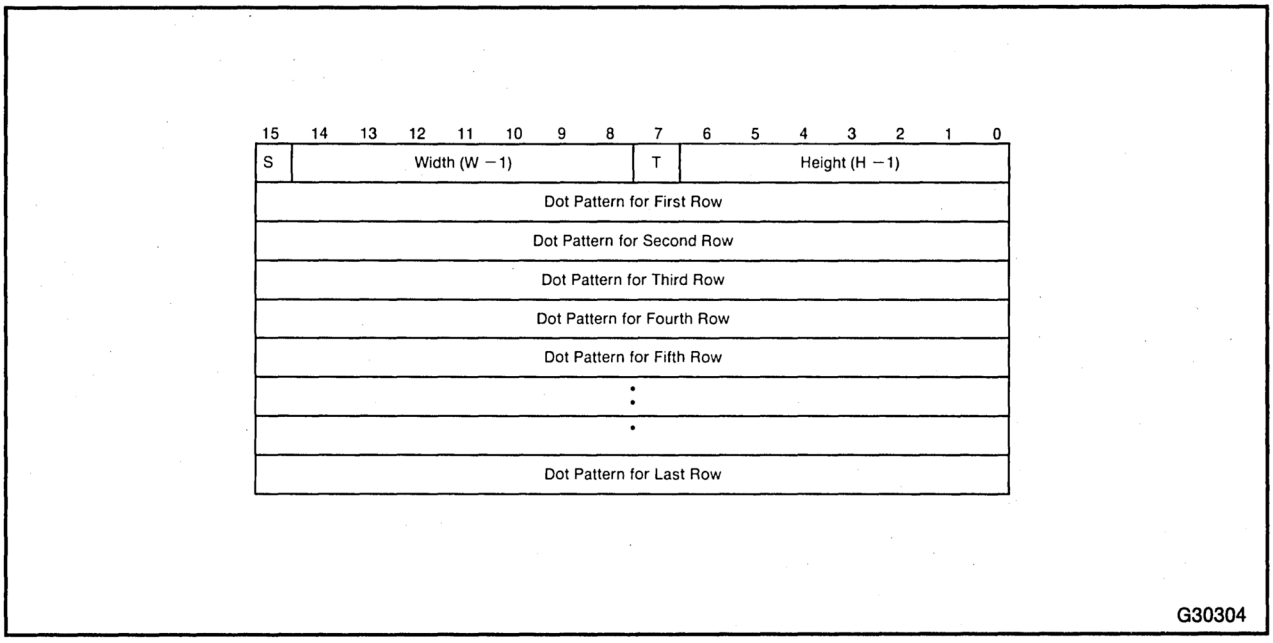


Figure 2-9. Character Descriptor Block Format

The S bit, the most significant bit of the Character Descriptor Block, indicates whether the Graphics Current Position Pointer (GCPP) is updated after the character is drawn. If the S bit is set to 1, the GCPP is not updated following character drawing. If the S bit is 0, the GCPP is updated following character drawing. Suppressing the updating of the GCPP after certain characters are drawn can be useful for underlining, overstriking, or creating other special effects.

The T bit (Trap bit), the most significant bit of of the first byte of the Character Descriptor Block can be set to initiate a software trap, which can be used if the character being drawn exceeds 16 pixels in either direction. The Trap bit can interrupt the CPU so that software can specially process a character with bitblt or other technique. For example, the Trap bit is useful for processing characters larger than 16×16 pixels. Such characters can be divided into quadrants and drawn by executing multiple character drawing commands. The Trap bit also can be used to provide elementary memory management for large fonts that may not reside entirely in physical memory at one time.

The height and width of the character refer to the difference between their limiting x and y coordinates. For example, a width of zero specifies a character one pixel wide and a height of zero specifies a character one pixel high.

The Char command (see Section 2.10.6) defines transparency/opaqueness, the character string pointer, and the number of characters in the string. The Character Descriptor Block containing the character to be drawn may be located anywhere in the 82786 memory space, which can be accessed with either an 8- or 16-bit reference to the specific character. The String Pointer specifies 8-bit (Byte Mode) or 16-bit (Word Mode) references for each character to be drawn. Standard character fonts can be drawn flexibly because path and rotation are defined with the Def_Char_Orient command (see Section 2.10.9) and inter-character spacing is defined with a Def_Char_Space command (see Section 2.10.11). This allows an application to specify variable spacing and direction of text, and rotate characters without altering the font. Simple one-bit per pixel character font definitions can be used in color applications because the Def_Color command specifies the foreground and background colors and the necessary bits are written for each pixel during the drawing process.

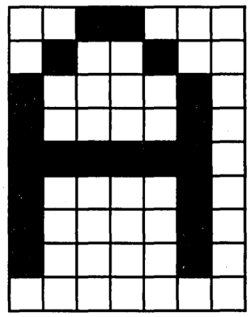
The Def_Char_Set command (see Section 2.10.10) specified 22-bit font base address and Char command specified parameter strings define the address of the Character Descriptor Block, which describes the character. Figure 2-10 illustrates the format of a character Descriptor Block for the character A.

In Byte Mode, characters are referenced in 8 bits, which allows existing software using ASCII character codes and EBCDIC to be converted to 82786-based systems. In Byte Mode, 256 words of the font are reserved for a lookup table. To locate a specific character in the table, the 8-bit Char command parameter string for the character is doubled (shifted left one bit) and then added to the 22-bit font base address pointer of the Def_Char_Set command. The resulting word address is the offset into the font table. The contents of the addressed word in the table are doubled and added to the font base address. This sum points to the starting address of the Character Descriptor Block, which describes the character. Figure 2-11 outlines Byte Mode, which permits only 256 characters in each 8-bit font. For details on the Character Descriptor Block, refer to the Def_Char_Set command in Section 2.10.10.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0
0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Character "A"

```
DW 8608H        ; S=1, T=0, Width (W - 1)=6, Height (H - 1)=8
DW 0011000B
DW 0100100B
DW 1000010B
DW 1000010B
DW 1111110B
DW 1000010B
DW 1000010B
DW 1000010B
DW 0000000B
```



G30304

Figure 2-10. Sample Character Descriptor Block

In Word Mode, the 16-bit Char command parameter string is doubled (shifted left one bit) before being added to the 22-bit Def_Char_Set font base address. The resulting sum is the starting address of the Character Descriptor Block (see Figure 2-12). Maximum Character Descriptor Block size is seventeen words of data, which supports approximately four thousand characters in one 16-bit font (worst case). Supplementary software using a lookup table can be used to access as many as 65,000 characters in a single font. Alternatively, for characters larger than native (16×16 pixels), the font can be organized as a bitmap and with the Bit_Blt_M command arbitrary sized single characters can be written.

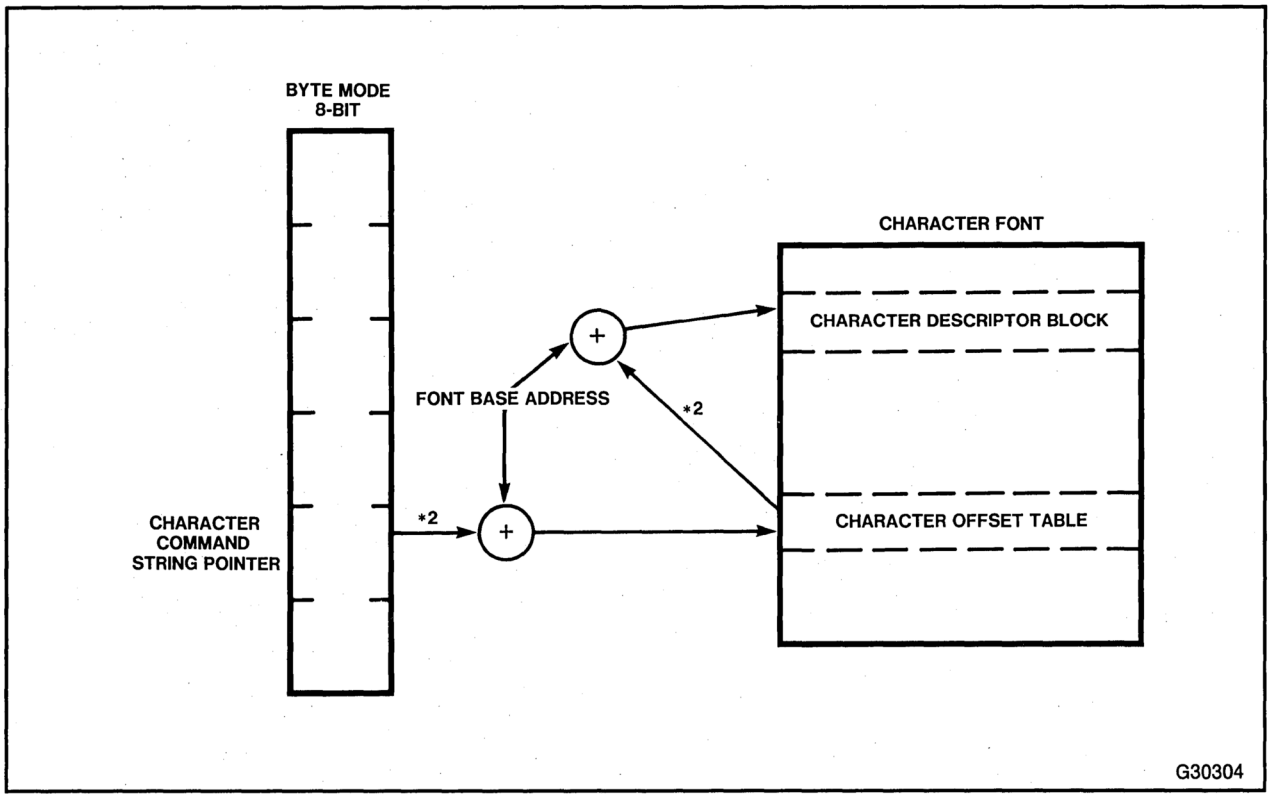


Figure 2-11. Byte Mode

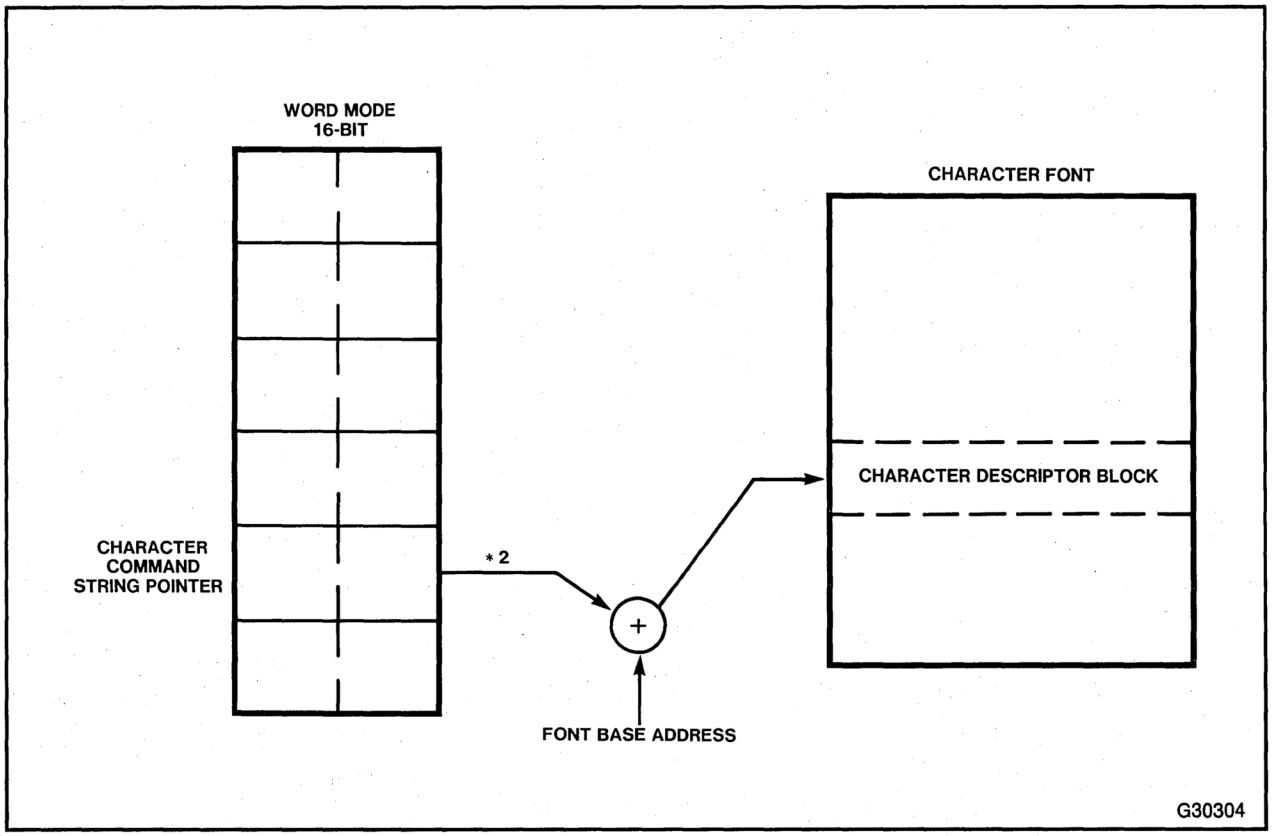


Figure 2-12. Word Mode

2.10 COMMAND DESCRIPTIONS

The following subsections describe Graphics Processor (GP) commands. The commands are organized in alphabetical order. When using any of these commands, program reserved bits to zero.

2.10.1 ABS_MOV–Move

Opcode 4FH

Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Opcode 4FH								Reserved							GECL
X coordinate															
Y coordinate															

Description: The Abs_Mov command moves the Graphics Current Position Pointer (GCPP) to the absolute location specified by the given X and Y coordinates. (See Figure 2-13.) The X and Y coordinates are specified in two's complement form and can be negative. The reserved bits should be programmed to zero.

Neither the Pick Successful Flag (GPSC), or the Bitmap Overflow Flags (GBMOV and GBCOV), in the Status Register are set as a result of this command, even if the GCPP moves beyond the clipping rectangle. For discussions of the clipping rectangle and PICK Mode, refer to Sections 2.6.4 and 2.6.5.

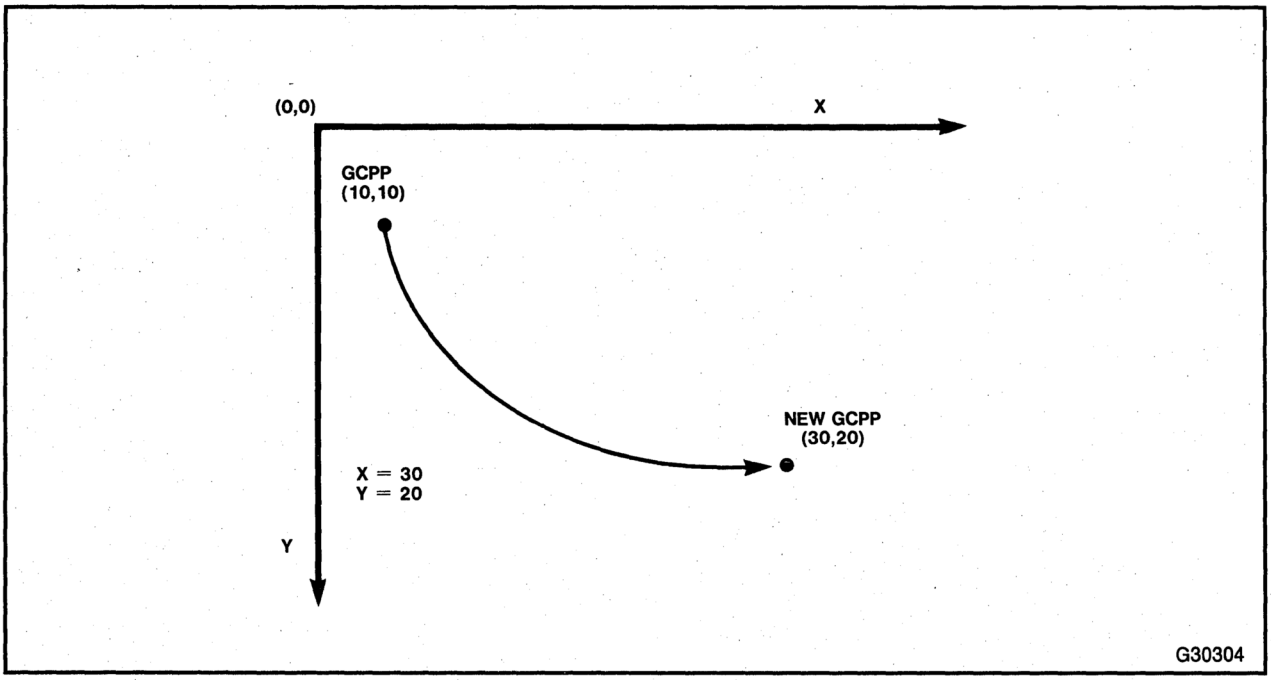


Figure 2-13. Absolute Move

2.10.2 ARC–Draw Arc

Opcode

68H - Arc Exclusion
69H - Arc Inclusion

Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Opcode 68H/69H							I/E	Reserved							GECL
dxmin															
dymin															
dxmax															
dymax															
radius															

Description

Two versions of the Arc command can be used: arc inclusion and arc exclusion. Each draws an arc with the center at the Graphics Current Position Pointer (GCPP) and the radius specified in the command. The parameters dxmin, dxmax, dymin, and dymax are the displacements (relative to the GCPP) for the endpoints of an inclusion/exclusion rectangle as shown in Figure 2-14. The inclusion rectangle includes the endpoints of the arc, but the exclusion rectangle does not. The I/E bit in the opcode determines which arc to draw. If the I/E bit equals 1, an inclusion arc is drawn. The inclusion arc lies inside the rectangle and includes points on the edge of the rectangle. If the I/E bit equals 0, an exclusion arc is drawn. Conversely, the exclusion arc lies outside the rectangle. The reserved bits should be programmed to zero.

The parameters are expressed as two's complement numbers and can be negative. However, if the radius is negative, no arc is drawn. If the radius is zero, the arc is reduced to a single point at the GCPP. If dxmin $>$ dxmax or dymin $>$ dymax, the inclusion/exclusion rectangle is treated as a NULL rectangle. No part of an arc is contained in a NULL rectangle.

The currently active texture, color, and logical operation are observed. The GCPP remains at the center when the command completes execution.

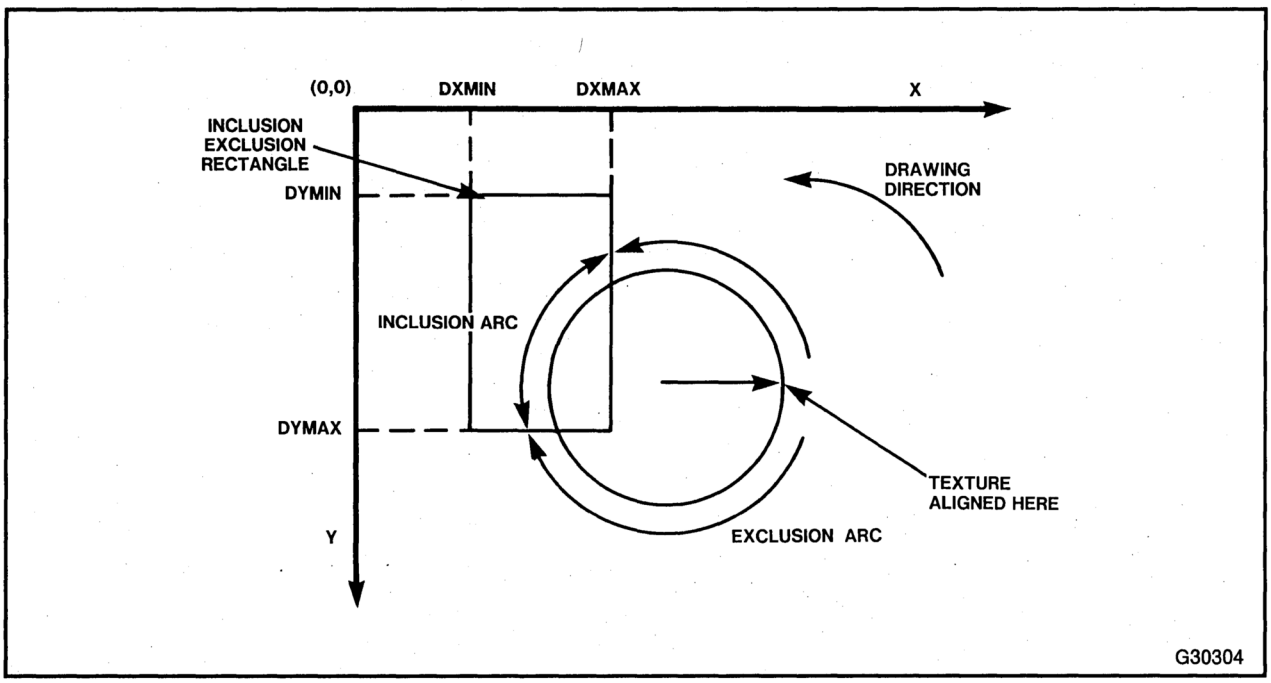


Figure 2-14. Draw Arc

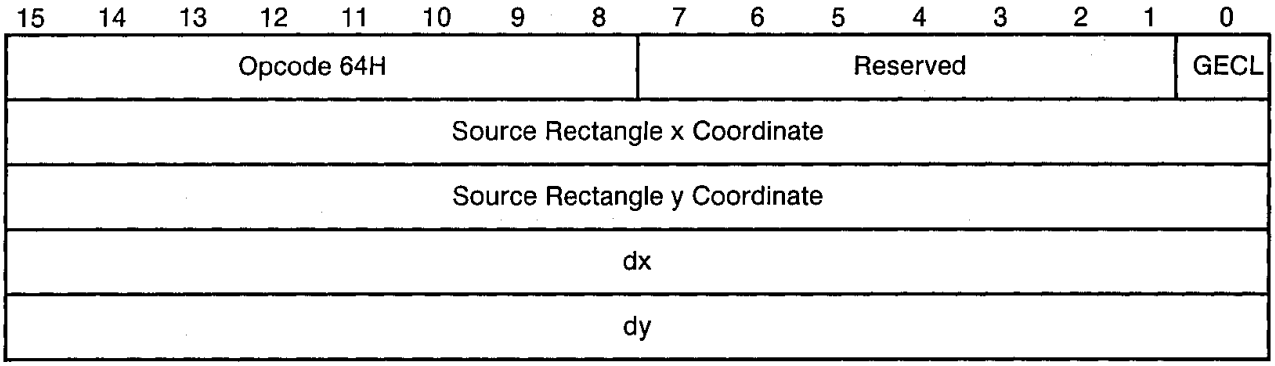
The clipping rectangle and the inclusion/exclusion rectangle are independent. Part of the inclusion/exclusion rectangle may lie outside the clipping rectangle. However, if any part of the arc falls outside of the clipping rectangle, the arc is drawn clipped and the Bitmap Overflow Flag (GBMOV) in the Status Register (GSTAT) is set. The arc is drawn in a counterclockwise direction with the texture aligned on a pixel basis to the point (xc + r, yc). The GCPP equals (xc,yc). For details on GSTAT, refer to Section 2.2.1.1 "Graphics Processor Status Register (GSTAT)."

PICK mode is not supported for this command. If drawing is in transparent mode, the texture must be solid (applies only to this command and the Circle command).

2.10.3 BIT_BLT–Bit Block Transfer

Opcode 64H

Format



Description

The Bit_Blt command copies a rectangular block of pixels within the current bitmap. The parameters specify the source rectangle with a corner at (x, y), width dx, and height dy. The source rectangle is copied to the destination such that one corner of the destination rectangle at coordinates (x, y) maps to the Graphics Current Position Pointer (GCPP), as shown in Figure 2-15. The height and width of both rectangles are the same. When the command completes execution, the GCPP moves to the location (GCPPx + dx, GCPPy).

The reserved bits should be programmed to zero. The parameters are expressed in two's complement form and can be negative. The (x, y) coordinates can be any of the four corners of the source rectangle, but corresponds only to the upper left corner if dx and dy are both non-negative. When dx = dy = 0, the single pixel at (x, y) is copied.

Each pixel of the destination rectangle is updated according to the currently defined logical operation. The Graphics Processor (GP) automatically adjusts the order of pixel transfer to accommodate source and destination rectangles that overlap. If the destination rectangle (not the source) crosses the clipping rectangle, no operation is performed and the the Bitmap Overflow Flag (GBCOV) in the Status Register (GSTAT) is set. When in PICK Mode, the Pick Successful Flag (GPSC) in GSTAT is set if any of the pixels in the destination rectangle lie within the clipping rectangle. Sections 2.6.4 and 2.6.5 discuss the clipping rectangle and PICK Mode. Refer to Section 2.2.1.1 "Graphics Processor Status Register (GSTAT)" for details on GSTAT.

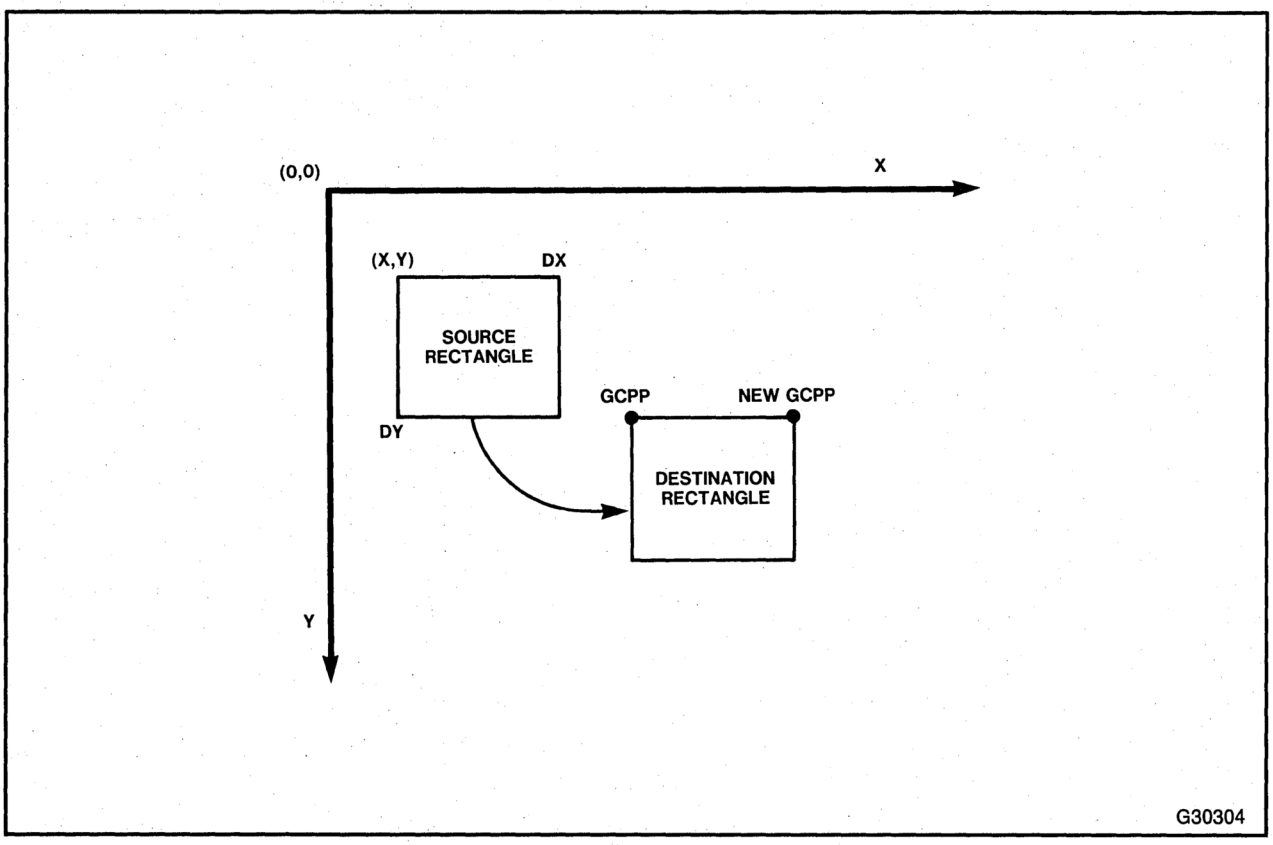


Figure 2-15. Bit Block Transfer

2.10.4 BIT_BLT_M–Bit Block Transfer Between Bitmaps

Opcode AEH

Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Opcode AEH								Reserved							GECL
Source Bitmap Lower Origin Address															
Reserved															
Source Bitmap Upper Origin Address															
Source xmax															
Source ymax															
Source rectangle x coordinate															
Source rectangle y coordinate															
dx															
dy															

Description

The Bit_Blt_M command copies a rectangular block of pixels across bitmaps. Unlike other bitblt commands, the source rectangle lies in a bitmap other than the currently active bitmap. The parameters specify the Source Bitmap Origin Address, which defines the source rectangle with the width xmax and height ymax. The Source Bitmap xmax and ymax parameters, as well as x, y, dx, and dy parameters are measured in pixels. The source rectangle defines a corner at (x, y), width dx, and height dy. Reserved bits should be programmed to zero.

The source rectangle is copied to the destination in the currently active bitmap (as defined by the previously specified Def_Bitmap command in Section 2.10.8) such that (x, y) maps to the Graphics Current Position Pointer (GCPP), as shown in Figure 2-16. The height and width of both rectangles are the same. If the original GCPP was (xc, yc), following command execution, the new GCPP location will be (xc + dx, yc).

The dx and dy parameters are expressed in two's complement form and can be negative. The (x, y) parameters correspond to the upper left corner if dx and dy are both non-negative, otherwise these parameters correspond to one of the other corners of the source rectangle. When dx = dy = 0, a single pixel at (x, y) is copied.

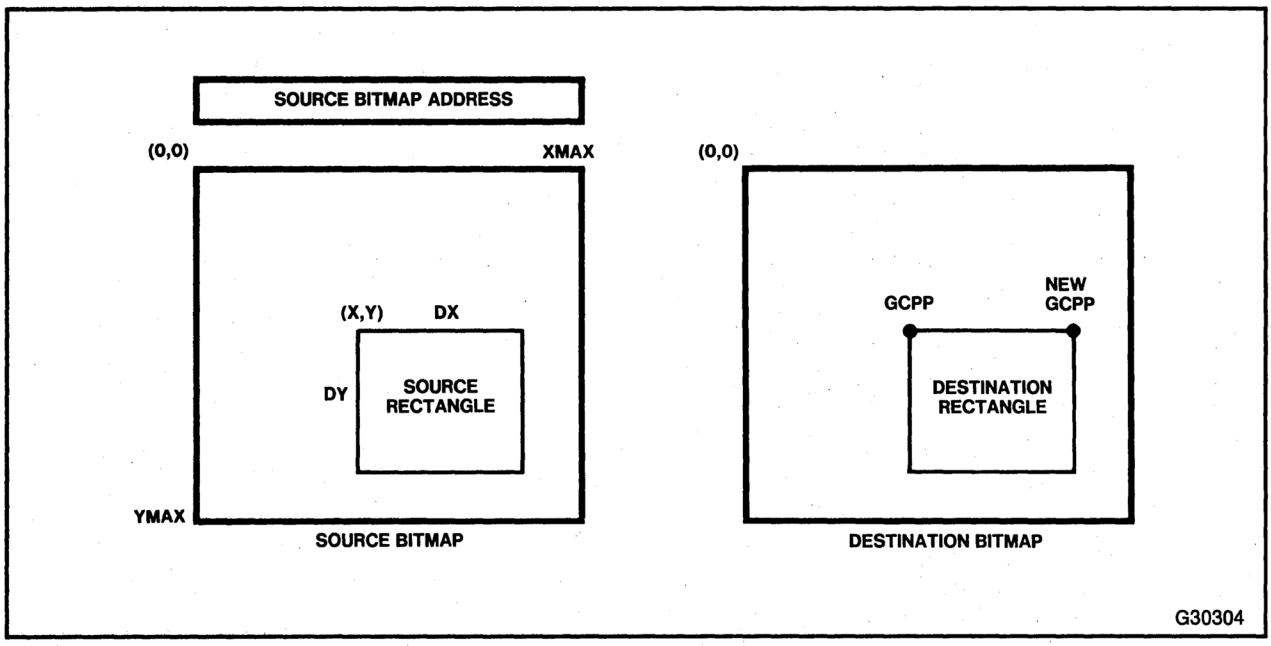


Figure 2-16. Bit Block Transfer Between Two Bitmaps

The Graphics Processor (GP) assumes the source bitmap has the same number of bits per pixel (bpp) as the destination. This transfers the requisite number of bytes from the source bitmap to the destination bitmap. Unexpected results can occur if the bpp for the source bitmap are not equal to the bpp of the destination.

Each pixel of the destination rectangle is updated according to the currently defined logical operation. If the destination rectangle crosses the clipping rectangle, no operation is performed and the Bitmap Overflow Flag (GBCOV) in the Status Register (GSTAT) is set. The source bitmap is not clipped.

In PICK Mode, the Pick Successful Flag (GPSC) in GSTAT is set if any pixels of the destination rectangle lie within the clipping rectangle. Sections 2.6.4 and 2.6.5 discuss the clipping rectangle and PICK Mode. For details on GSTAT, refer to Section 2.2.1.1 "Graphics Processor Status Register (GSTAT)."

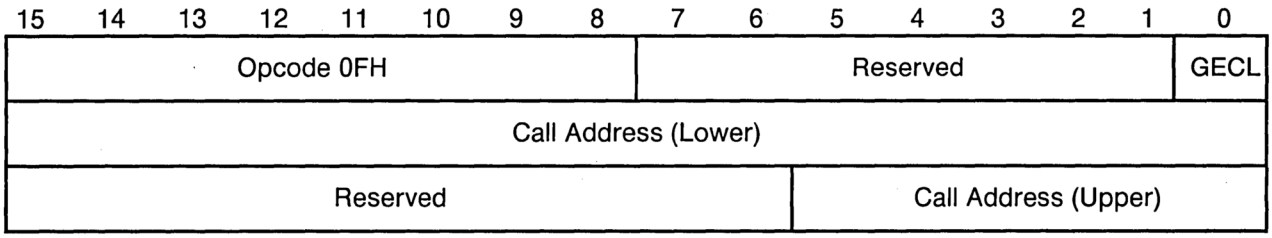
NOTE

The Bit_Blt_M command defines the source bitmap and the last specified Def_Bitmap command (defined in Section 2.10.8) defines the active bitmap, which is the destination. The bits per pixel (bpp) of the source and destination bitmaps must be the same.

2.10.5 CALL–Call Subroutine

Opcode 0FH

Format



Description

The Call command calls a subroutine of Graphics Processor (GP) commands. When the GP encounters a Return command in the subroutine, control returns to the command following the Call command. With Call and Return commands, you can build subroutines, nested to an arbitrary depth. Reserved bits should be programmed to zero.

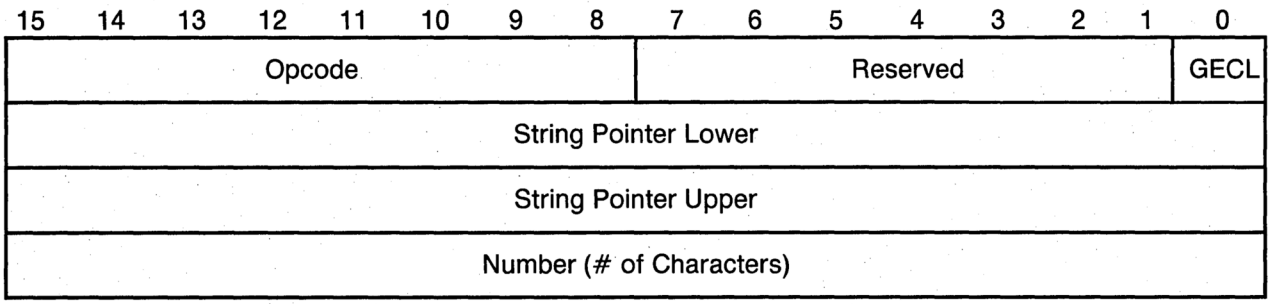
Before issuing Call commands, the Graphics Processor Stack Pointer (GSP Control Register), must be initialized. Use the Load_Reg command to load the GSP with the stack address. Following an exception which prematurely terminates the GP command stream, be sure to restore or reinitialize the GSP. The GP stack grows toward lower addresses.

After receiving a Call command, the GP increments the current Instruction Pointer (GCIP) by 6 (to point to the following command) and pushes the value onto the stack. Then, the GCIP is loaded with the subroutine address specified by the Call command. Control jumps to the called subroutine. Each subroutine call uses four bytes of stack space, which causes the GSP to be decremented by 4 each time a subroutine call occurs.

2.10.6 CHAR–Draw Character String

Command	Opcode
CHAR_OPAQUE	A600
CHAR_TRANSPARENT	A700
CHAR_RV_OPAQUE	A800
CHAR_RV_TRANSPARENT	A900

Format



Description

This command writes a string of characters into the active bitmap. The character dot matrix is read out of the active character table and the characters are written into the active bitmap using the active character path, character rotation and inter-character spacing. Reserved bits should be programmed to zero. The characters can be drawn transparent or opaque and reverse video. If normal video and transparent (opcode A700), then a "1" in the character descriptor dot matrix specifies the active foreground color and a "0" specifies no action. If opaque (opcode A600), a "1" specifies the active foreground color and a "0" the active background color.

If reverse video and transparent (opcode A900), then a "0" in the character descriptor dot matrix specifies the active foreground color and a "1" specifies no action. If reverse video and opaque (opcode A800), a "1" specifies the active background color and "0" specifies the active foreground color. Again the characters are drawn only in the planes specified by the currently active Mask and using the currently defined logical operation.

Each character in a character font has an independently programmable size of up to 16×16 pixels, which allows individual characters to be different sizes for proportional spacing. Information about each character resides in memory and is provided by the user. Starting from an even byte address, the character information is stored in consecutive words of memory forming a character block containing n+1 words of memory, where n is the pixel height of the character. Each Character Descriptor Block has the format shown in Figure 2-17.

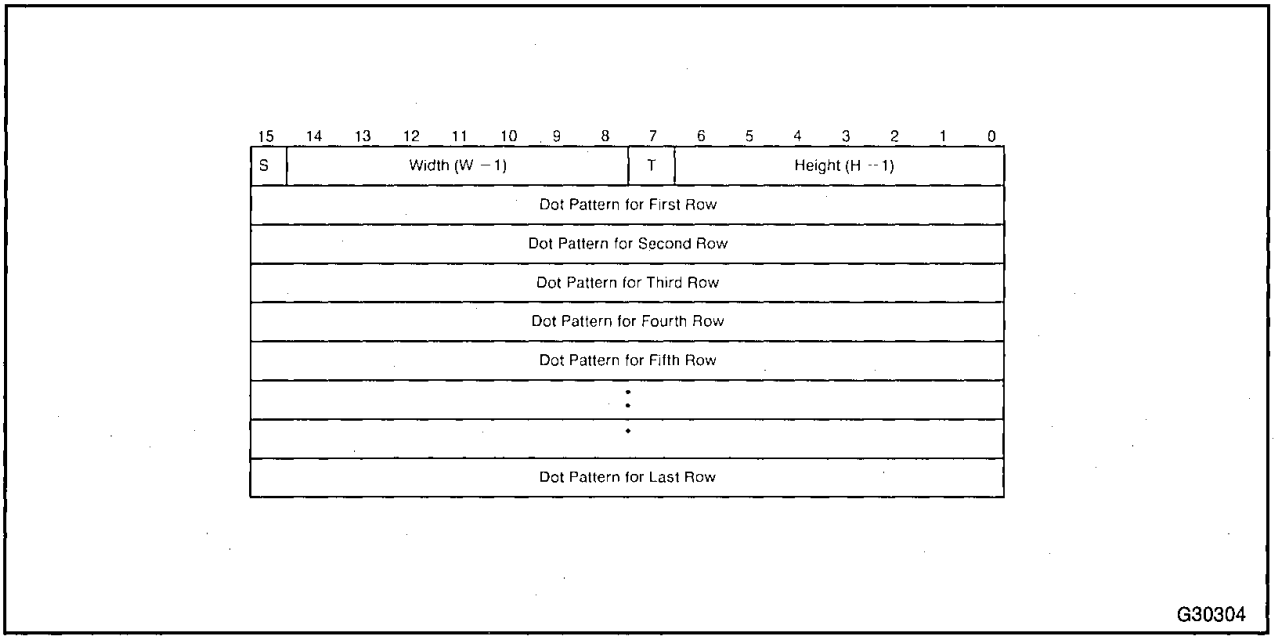


Figure 2-17. Character Descriptor Block

Each character is identified by specifying the address of the starting block. This address is an offset from the character table base address, which is programmed before using the Def_Character_Set command.

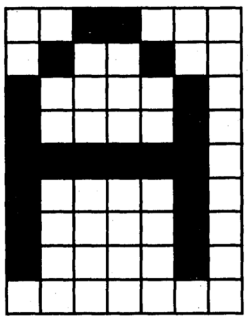
Each character block must start at an even byte address. The dot patterns for each row must reside in consecutive words in memory. The first word defines the height and width of the character (see Figure 2-17) and whether the character is an overstrike or exceeds the 16×16 pixel limit in either direction. If the dot pattern for the width of the character is less than 16 pixels, the dot pattern for the row must be right justified in the word of the Character Descriptor Block. However, the dot pattern defining the character in the character cell can be left justified as shown in Figure 2-18. In this way, a character can create its own proportional spacing.

The S bit, the most significant bit of the Character Descriptor Block, indicates whether the Graphics Current Position Pointer (GCPP) is updated as defined by the Def_Char_Orient command after the character is drawn. If the S bit is set to 1, the GCPP is not updated following character drawing. If the S bit is 0, the GCPP is updated following character drawing. Suppressing the updating of the GCPP after certain characters are drawn can be useful for underlining, overstriking, or creating other special effects.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0
0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Character "A"

```
DW 8608H        ; S=1, T=0, Width (W -1)=6, Height (H -1)=8
DW 0011000B
DW 0100100B
DW 1000010B
DW 1000010B
DW 1111110B
DW 1000010B
DW 1000010B
DW 1000010B
DW 0000000B
```



G30304

Figure 2-18. Sample Character Descriptor Block

The T bit (Trap bit), the most significant bit of the first byte of the Character Descriptor Block can be set to initiate a software trap, for specialized processing of a character that exceeds 16 pixels in either direction or elementary memory management. The Trap bit can interrupt the CPU so that software can specially process a character with bitblt or other technique. A character larger than 16×16 pixels can be divided into quadrants and drawn by executing multiple character drawing commands. The Trap bit also can be used to provide elementary memory management for large fonts that may not reside entirely in physical memory at one time.

The height and width of the character refer to the difference between their limiting x and y coordinates. For example, a width of zero specifies a character one pixel wide and a height of zero specifies a character one pixel high.

Before issuing the Char command, the current character font must be established by using the Def_Char_Set command. This command also determines whether to use Word or Byte Mode to select characters from the font.

The initial x,y coordinates are taken to be the GCPP position. If the character is not specified to be non-spacing then, after writing each character, the GCPP is moved to a point "character width + space − 1" away (in the direction of the character path) from the initial point. The new GCPP position becomes the initial point for the next character. The character boundary is assumed to be the bounding rectangle of the character. If any of the characters cross the clipping rectangle, the bitmap overflow (GBCOV) flag is set and only the portions of the character lying within the clip rectangle are drawn. The GCPP is updated to lie at the final position of the Nth character and is unaffected by clipping of the characters.

The character string is interpreted to be either a string of bytes or a string of words depending upon the mode set up by the Def_Char_Set command. In Word Mode, the string pointer parameter of the Char command points to a string of n characters. The string is interpreted as shown in Figure 2-19.

In this mode, each word in the character strings corresponds to a character from the defined font. The correspondence is defined as follows. The word code is added to the Character-font's base address (22-bits) and the resulting 22-bit quantity serves as the address to the beginning of the character description block. Figure 2-20 shows the operation of the Char command in Word Mode with the character path in the standard left to right orientation.

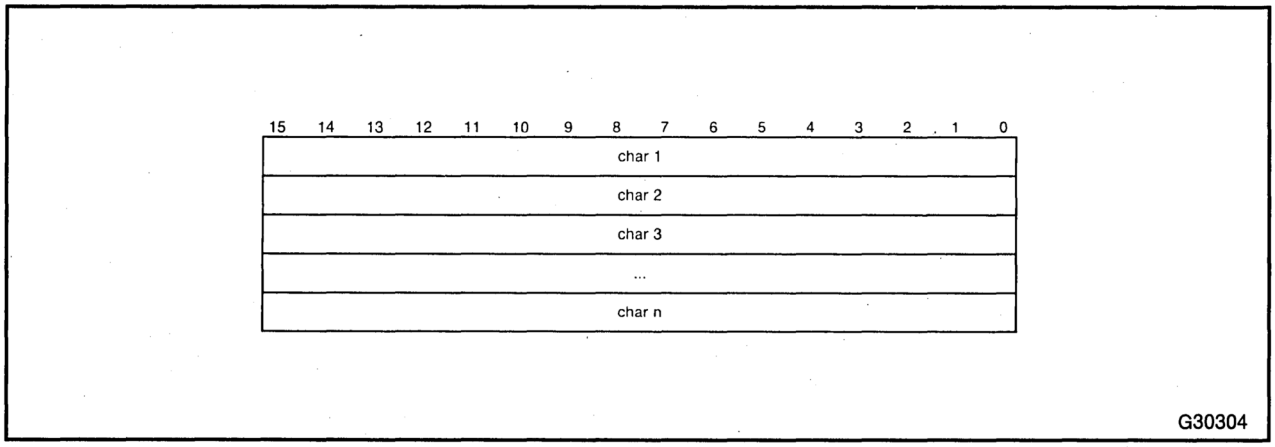


Figure 2-19. Word Mode Selector Words

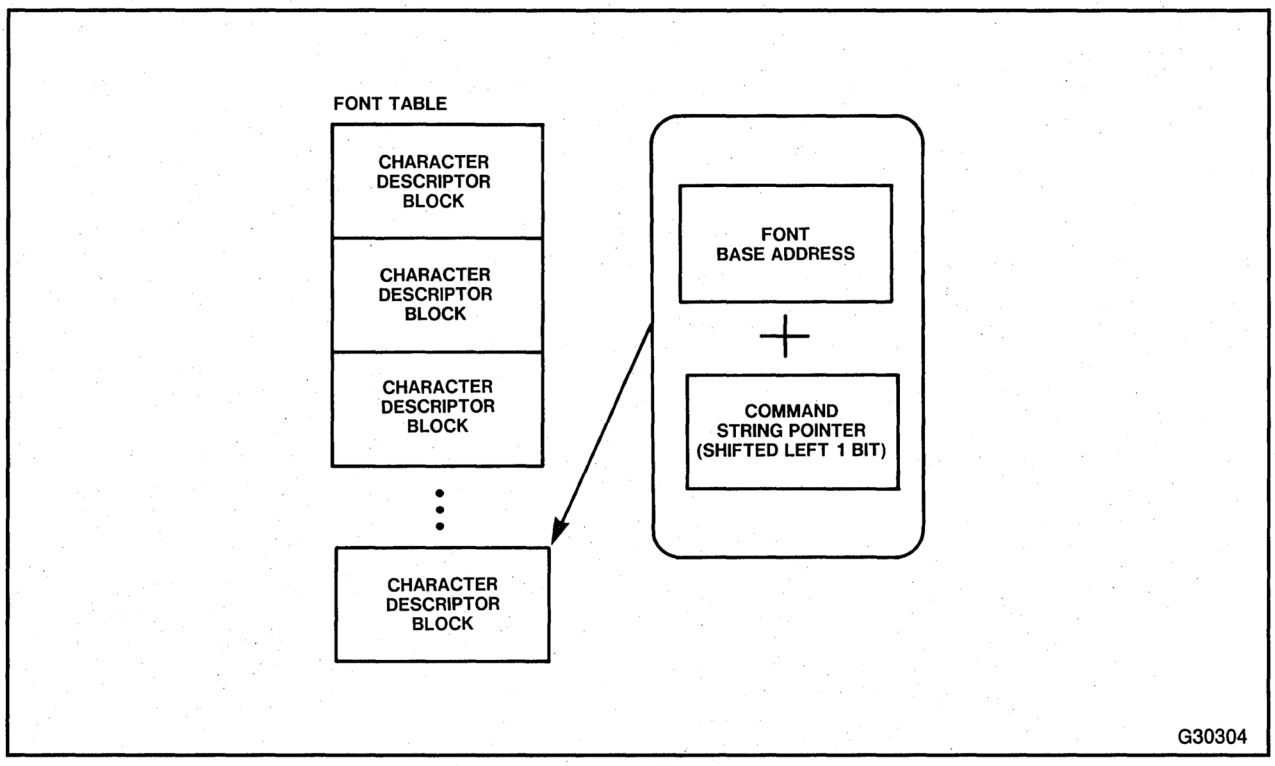


Figure 2-20. Character Command in Word Mode

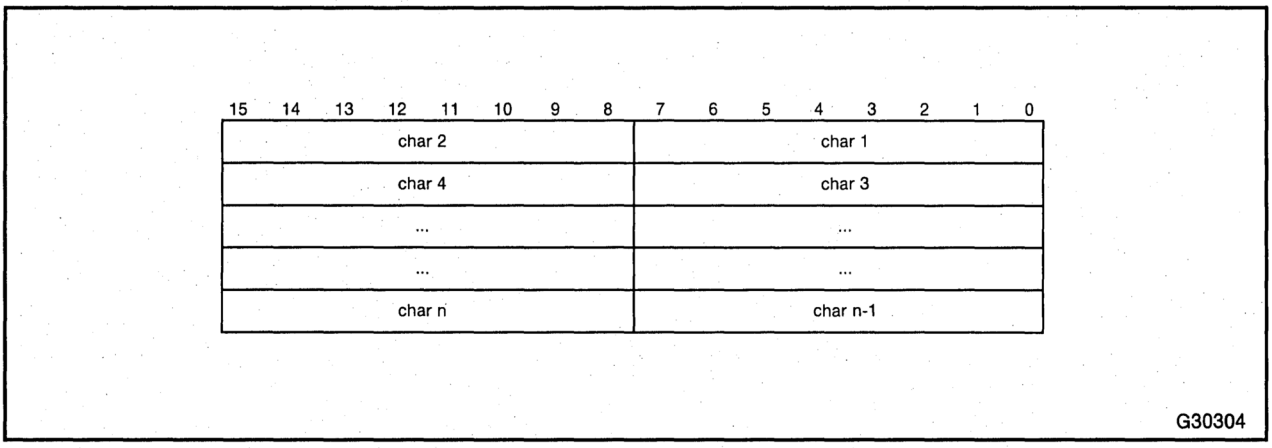


Figure 2-21. Byte Mode Selector Words

If in byte mode, the character string is interpreted as shown in Figure 2-21.

In this mode, each byte in the character string corresponds to a character from the character font. One more level of indirection is added to the character address generation. The byte is added to the Character-font's base address and the resulting 22-bit quantity addresses a word from the memory. This word is added to the character-font's base address and the resulting 22-bit address points to the beginning of the character descriptor block. The net effect is that of a lookup table being present at the head of the character font itself.

If any of the characters are defined with a TRAP bit set, all further processing of the string is aborted and the GTRP bit is set in the status register, which might cause an interrupt depending upon the contents of the Interrupt Mask Register or may cause an onset of the POLL mode depending upon the POEM register. In case of the Char command being aborted because of a GTRP bit, further information can be gotten from reading registers GCNT and GCIP. GCNT holds the number of characters remaining to be drawn when the GTRP bit was encountered while GCIP holds, as always, the address of the command that caused the character string to be drawn.

If the PICK mode is on, the GPSC flag in the status register is set if any of the pixels lying within the character bounding rectangle also lie in the clipping rectangle. In case of the PICK mode or normal mode, the Char command is processed for the entire character string. For instance, even if the first character overflows clipping rectangle boundaries, the rest of the character string is scanned for other characters that may lie within the clipping boundary. Further information is accessible via registers GCNT and GCIP. GCNT holds the number of characters that were remaining to be drawn when the last exception occurred while GCIP holds the address of the Char command that was under execution.

Note that the character string for all character drawing must begin on an even byte address. Byte strings that are aborted with a trap may require special handling to restart on an even byte address.

Character clipping is supported for 16-bit character codes only. In addition, if the top of the characters are to be clipped, the top row of pixels of the character must begin on an even numbered scan line (scan lines are numbered from top to bottom beginning with number 0). This restriction does not apply to characters which have only the sides or bottom clipped or which are not clipped at all.

2.10.7 CIRCLE–Draw Circle

Opcode 8EH

Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Opcode 8EH								Reserved							GECL
Radius															

Description The Circle command draws a circle with the center at the Graphics Current Position Pointer (GCPP) and radius as specified by the command (see Figure 2-22). The GCPP remains at the center of the circle. Pixels are updated according to the current texture and logical operation. A circle with a radius set to zero consists of the single point at the GCPP. If the radius is negative, the circle is not drawn. If the circle crosses the clipping rectangle, the circle is drawn clipped and the Bitmap Overflow Flag (GBMOV) in the Status Register (GSTAT) is set.

Reserved bits should be programmed to zero.

PICK Mode is not supported for this command. If drawing is in transparent mode, the texture must be solid (applies only to this command and Arc command).

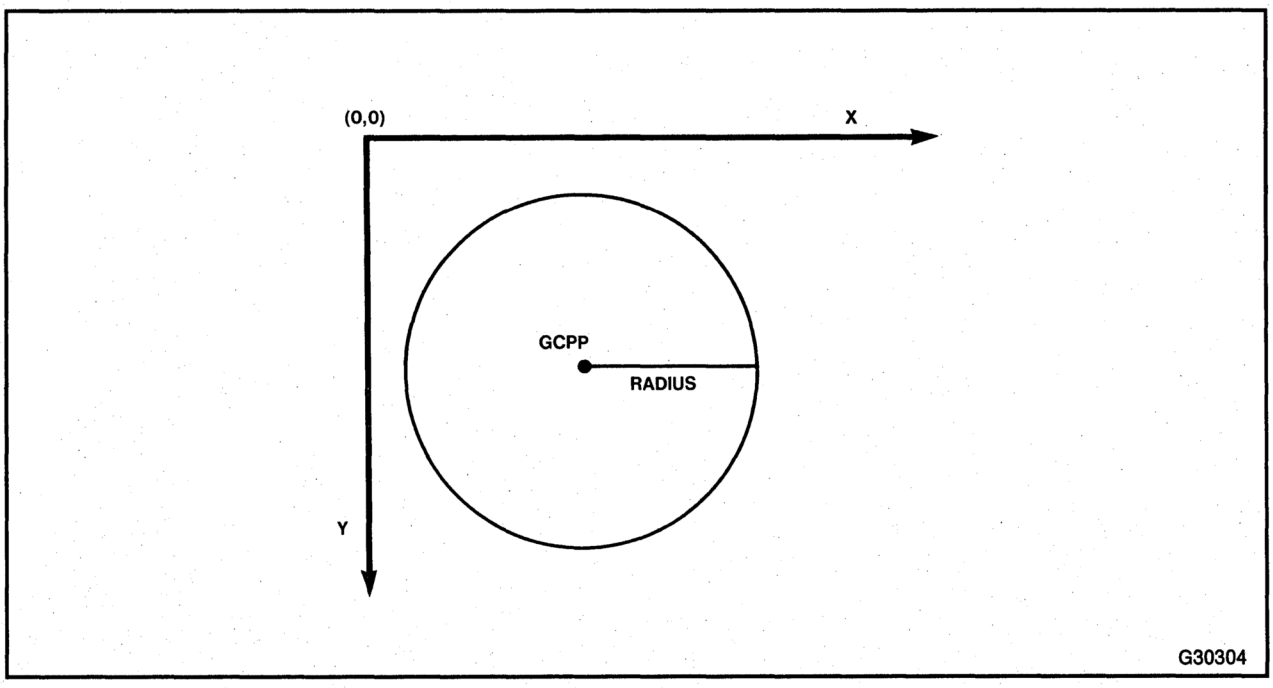


Figure 2-22. Draw Circle

2.10.8 DEF_BITMAP–Define Bitmap

Opcode 1AH

Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Opcode 1AH								Reserved							GECL
Bitmap Lower Origin Address															
Reserved										Bitmap Upper Origin Address					
xmax															
ymax															
bpp (1,2,4,8)															

Description

The Def_Bitmap command defines the active bitmap, which continues to be active until another Def_Bitmap command is executed. Def_Bitmap defines the bitmap as the rectangle spanned by the point (0,0) at the Origin Address in the top left corner and (xmax, ymax, which are specified in pixels) in the bottom right corner (see Figure 2-23). Bitmaps must begin at a word (even byte) address. The bitmap is activated with the Graphics Current Position Pointer (GCPP) at the Origin Address (0,0); the clipping rectangle is set to the bitmap boundary until the Def_Clip_Rect command redefines it. Reserved bits should be programmed to zero.

The bpp parameter sets the number of bits per pixel for the bitmap and must be 1, 2, 4, or 8. If any other value is used, if ymax < 0, if xmax does not satisfy the equation below, or if xmax exceeds 32K-1, the Illegal Bitmap Definition Flag (GIBMD) in the Status Register (GSTAT) is set, which may cause an interrupt depending on the Interrupt Mask Register (GIMR) discussed in Section 2.3.4 "Exception Handling." If an illegal bpp value is specified or ymax < 0, the bpp defaults to 1.

A bitmap must be an integral number of words wide. The value for xmax must satisfy the following equation:

$$[(xmax + 1) * bpp] \text{ MOD } 16 = 0$$

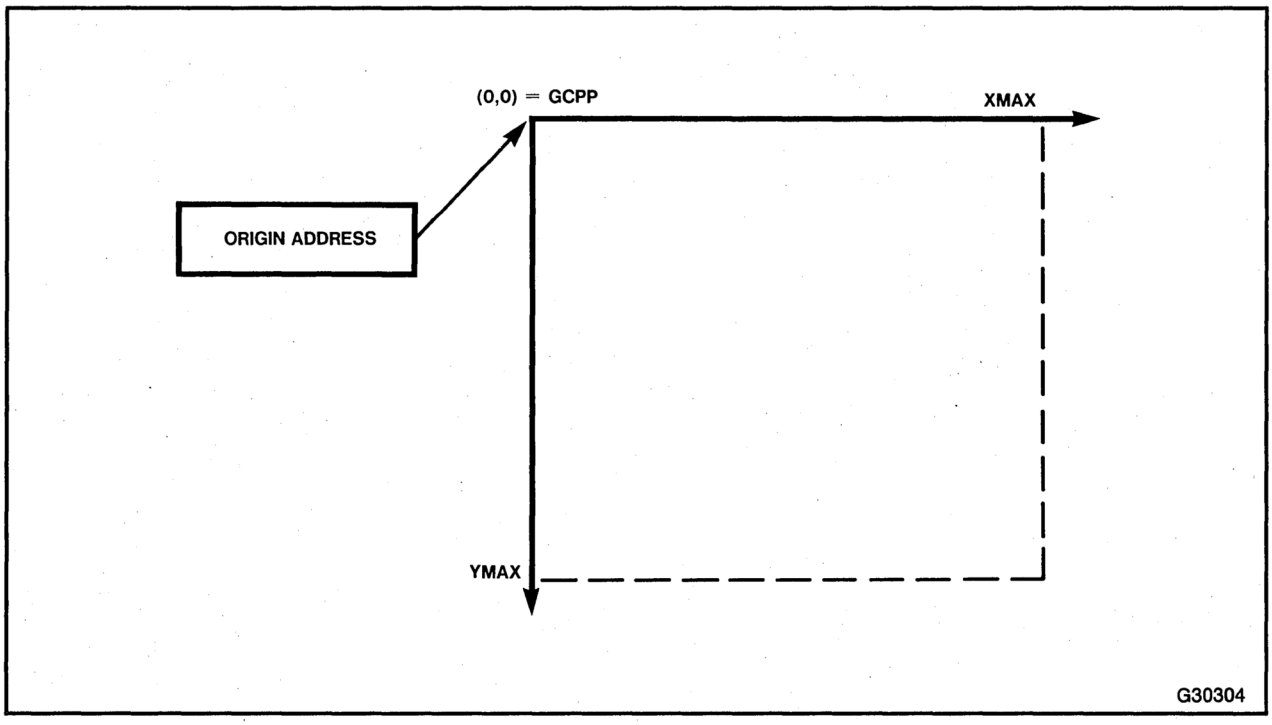


Figure 2-23. Define Bitmap

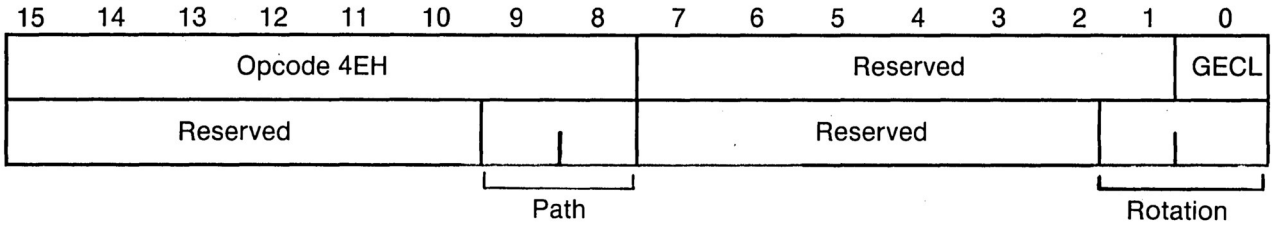
For example, with an xmax value of one (xmax = 1), the bitmap consists of two pixels: x=1, x=0. The ymax parameter does not have a similar restriction. However, if xmax or ymax are negative, the results are unspecified. The absolute maximum value for xmax and ymax is 7FFF. If a value greater than 7FFF is specified for either xmax or ymax, xmax or ymax defaults to zero.

No default definition of a bitmap exists. Thus, after RESET, only nondrawing commands may be executed prior to the Def_Bitmap command. Drawing commands issued before establishing a bitmap with the Def_Bitmap command have unspecified results.

2.10.9 DEF_CHAR_ORIENT–Define Character Orientation

Opcode 4EH

Format



Description The Def_Char_Orient command sets the character orientation used by the Char command. The path parameter defines the direction to update the Graphics Current Position Pointer (GCPP) after each character is drawn. The rotation parameter sets the orientation of each character. Specify both parameters as degrees counterclockwise to the horizontal axis. Parameters must be selected from the following path and rotation values:

Path and Rotation Values	Degrees
00	0
01	90
10	180
11	270

The rotation defined by this command remains active until redefined by another Def_Char_Orient command. Figure 2-24 shows possible character orientations. Reserved bits should be programmed to zero.

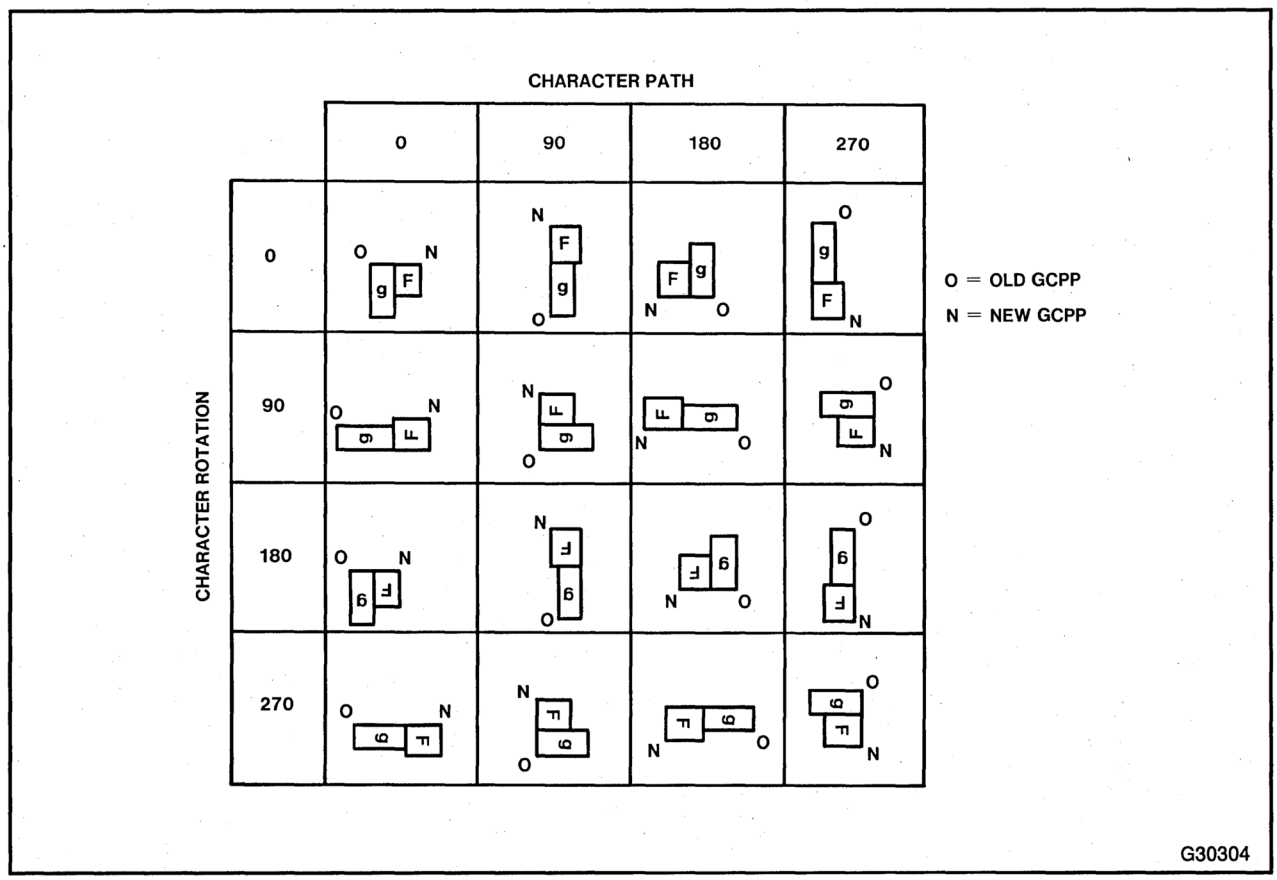


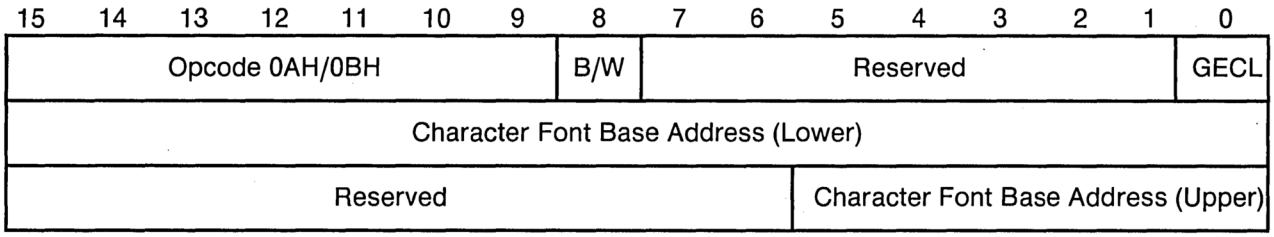
Figure 2-24. Define Character Orientation

2.10.10 DEF_CHAR_SET–Define Character Set

Opcode

0AH - Word Mode
0BH - Byte Mode

Format



Description

The Def_Char_Set command establishes the active character font used for drawing characters in the bitmap. After Def_Char_Set establishes the font, use the Char command to draw character strings into the bitmap. No default character font exists; you must use Def_Char_Set before drawing characters.

The command parameters specify the base address for the font, which must be an even byte address. The font is then addressed in one of two modes: Byte or Word. The opcode B/W bit selects the mode. Reserved bits should be programmed to zero.

Each character is described by a Character Descriptor Block. Figure 2-25 shows the layout of a Character Descriptor Block with the upper left corner placed at the GCPP. The placement of the GCPP changes as specified by the S bit in the Char command and the path and rotation in the Def_Char_Orient command.

Each Character Descriptor Block must start at an even byte address and the dot patterns for each row are stored in consecutive words of the Character Descriptor Block. The height and width of each character cannot be more than 16 pixels. When the character width is less than 16, the dot pattern for each row must be stored right justified within the word.

The first word of a Character Descriptor Block specifies the height and width of the character. These values are stored as the Height minus one (Height −1) and Width minus one (Width −1). Different characters in the same font can have different dimensions. You must ensure that the size of each character matches the number of rows used and that dot patterns are right justified within the word. Although, the character can be left justified within the character cell as shown in Figure 2-26.

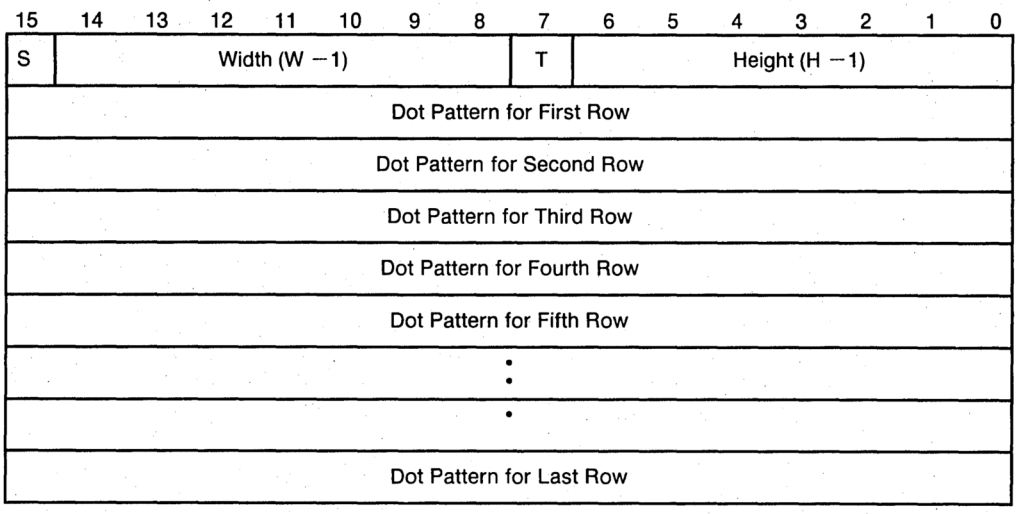


Figure 2-25. Character Descriptor Block

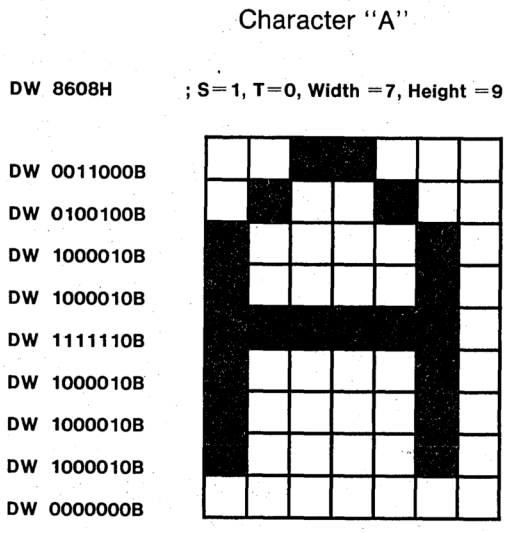


Figure 2-26. Sample Descriptor Block

The first word of the Character Descriptor Block also contains S and T bits for the character. The S bit determines whether or not the Graphics Current Position Pointer (GCPP) is updated after the character is drawn. When S is set to the value zero, the following characteristics determine how the GCPP is updated:

- Character rotation as defined by the Def_Char_Orient command in Section 2.10.9
- Character width/height as specified in the Character Descriptor Block of the character font
- Intercharacter spacing occurs

When the S bit value is one, the GCPP is not updated, which can be useful when underlining or drawing accented characters.

The T bit, or Trap Bit, when set to 1, causes the Graphics Processor (GP) to trap during character drawing operations. When the GP encounters a set T bit, the Character Trap bit (GCTP) in the Status Register (GSTAT) is set, which can trigger an interrupt or cause the GP to enter Poll State based on the Interrupt Mask (GIMR) and/or Poll On Exception Mask (GPOEM). For details on the GIMR and GPOEM, refer to Section 2.3.4 "Exception Handling."

The T bit can be used for characters that require special handling. For example, it can trigger an interrupt to the CPU to initiate emulation of characters that are greater than 16×16 pixels or provide elementary memory management to handle large character fonts that may not reside entirely in physical memory.

Figure 2-26 shows a sample Character Descriptor Block representing the character A. In the example, the starting GCPP is at the upper left corner, the Trap Bit (T) is turned off and the S bit is set to 1, which means the GCPP is not updated after the character is drawn. The width is set to 6 and height is set to 8, which indicates the character cell size is 7×9 pixels. The Character Descriptor Block data for each row is right justified in the word in which it is stored (as denoted by the Define Word notation DW). However, the character is left justified within the character cell to provide a 1-bit column of intercharacter spacing.

The opcode B/W bit of the Def_Char_Set command selects the character addressing mode. In Word Mode, a character string is represented by a string of consecutive words. Each word is the displacement from the character font base address of the pointer to the Character Descriptor Block for that character. Character addressing in Word Mode is illustrated in Figure 2-27. The Character Selector Word is doubled (shifted left one bit) and added to the base address to locate the Character Descriptor Block.

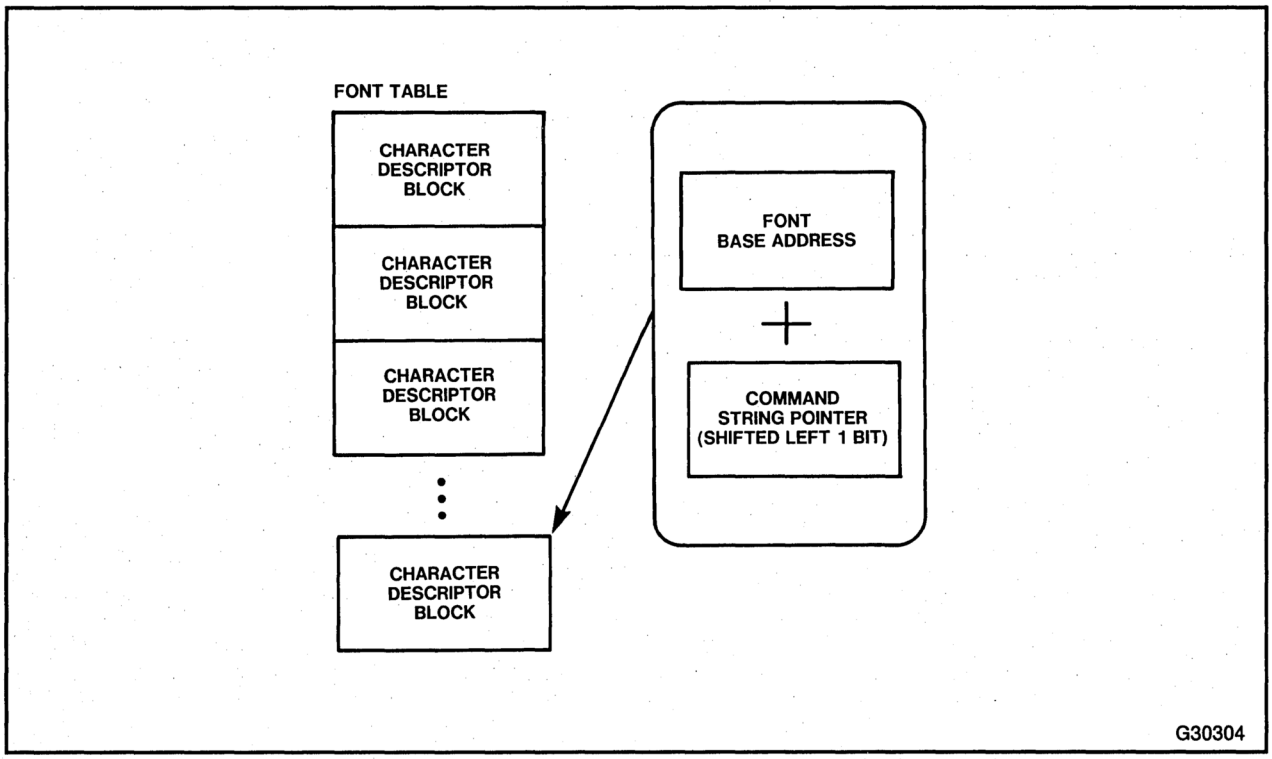


Figure 2-27. Word Mode Character

In Byte Mode, a character string is represented by a string of consecutive bytes. The 82786 computes the location of the Character Descriptor Block for a specific character while executing a Char command by shifting the Character String Pointer left one bit before adding it to the font base address. The result is an address into the offset table. The 82786 shifts the contents of the addressed word in the offset table left one bit, then adds it to the font base address. The resulting sum yields the starting address of the Character Descriptor Block. This establishes a character lookup table, located at the start of the font table. Figure 2-28 illustrates Byte Mode character addressing.

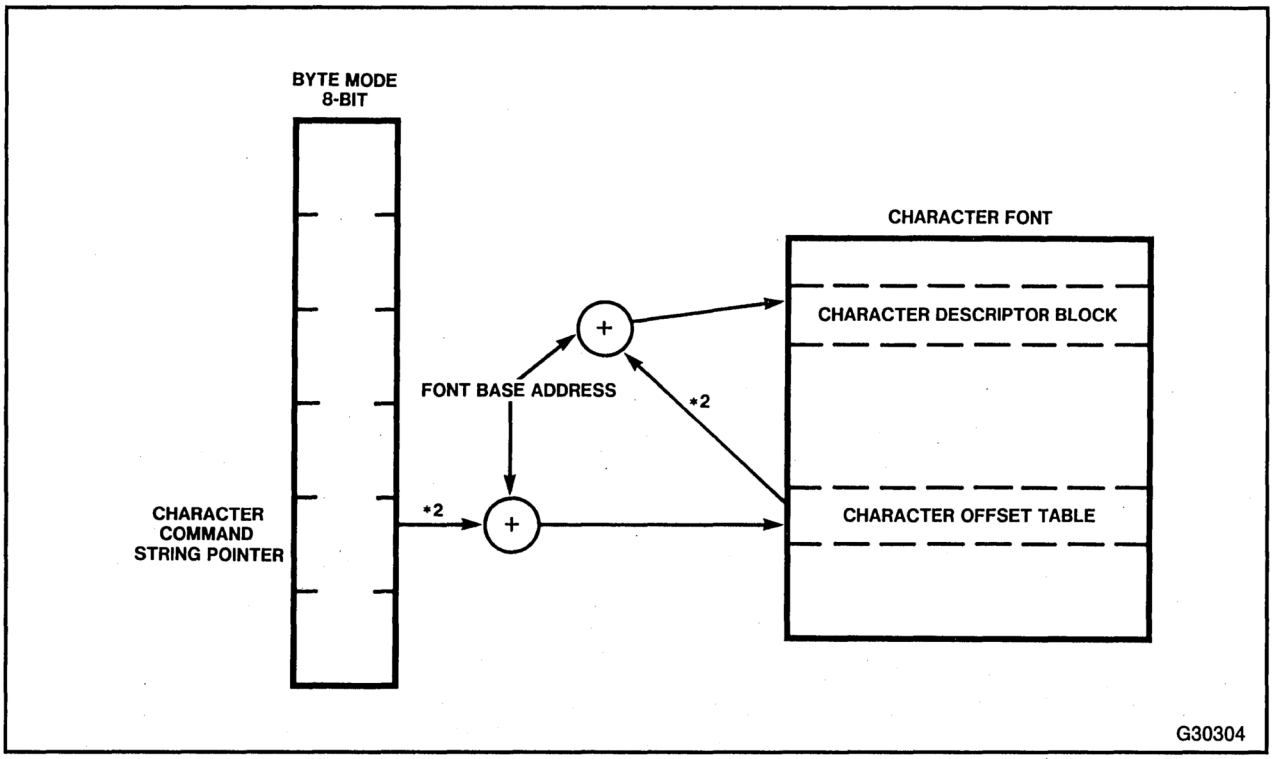
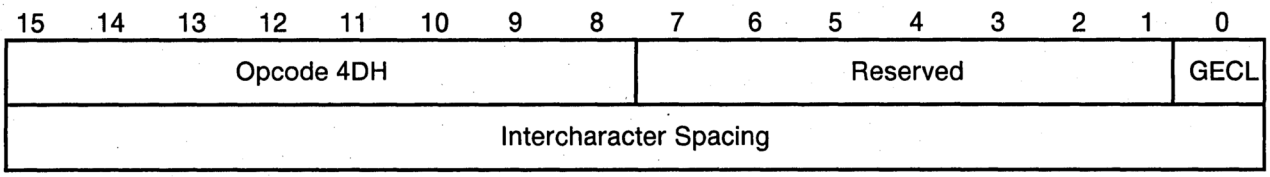


Figure 2-28. Byte Mode Character

2.10.11 DEF_SPACE–Define Space

Opcode 4DH

Format



Description The Def_Space command sets the intercharacter spacing, in pixels, which subsequent Char commands require. Def_Space also defines GCPP positioning along the X-axis for all bit block transfer (bitblt) operations initiated by a BitBlt, BitBlt_M, or BitBlt_E command. For example, with initial GCPP coordinates (x,y), the new GCPP location after executing a bitblt command becomes (x + dx + space, y). No default spacing exists. Following RESET (see Section 2.3.3 "RESET and Initialization"), spacing is undefined. Spacing set by this command remains active until changed by another Def_Space command.

The intercharacter spacing is always in the direction of the character path defined by the Def_Char_Orient command, described in Section 2.10.9. The intercharacter spacing parameter is a two's complement number and can be negative, which may be used by italic fonts and other kerning applications. Reserved bits should be programmed to zero.

2.10.12 DEF_CLIP_RECT–Define Clipping Rectangle

Opcode 46H

Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Opcode 46H								Reserved							GECL
xmin															
ymin															
xmax															
ymax															

Description

The Def_Clip_Rect command defines the current clipping rectangle for the active bitmap against which all subsequent drawing commands are clipped. Reserved bits should be programmed to zero. The xmin, ymin, xmax, and ymax parameters cannot be negative numbers, and must lie within a defined bitmap. A drawing operation updates only those pixels that fall on or within the clipping rectangle that lies within a defined bitmap. If xmax $<$ xmin, or ymax $<$ ymin, no updates of the bitmap are possible.

After a Def_Bitmap command, the default clipping rectangle is the entire bitmap. The Def_Clip_Rect command sets the clipping rectangle to the unique rectangle spanning the two corners (xmin, ymin) and (xmax, ymax) as shown in Figure 2-29. The four parameters are two's complement numbers, which must be entered as positive numbers.

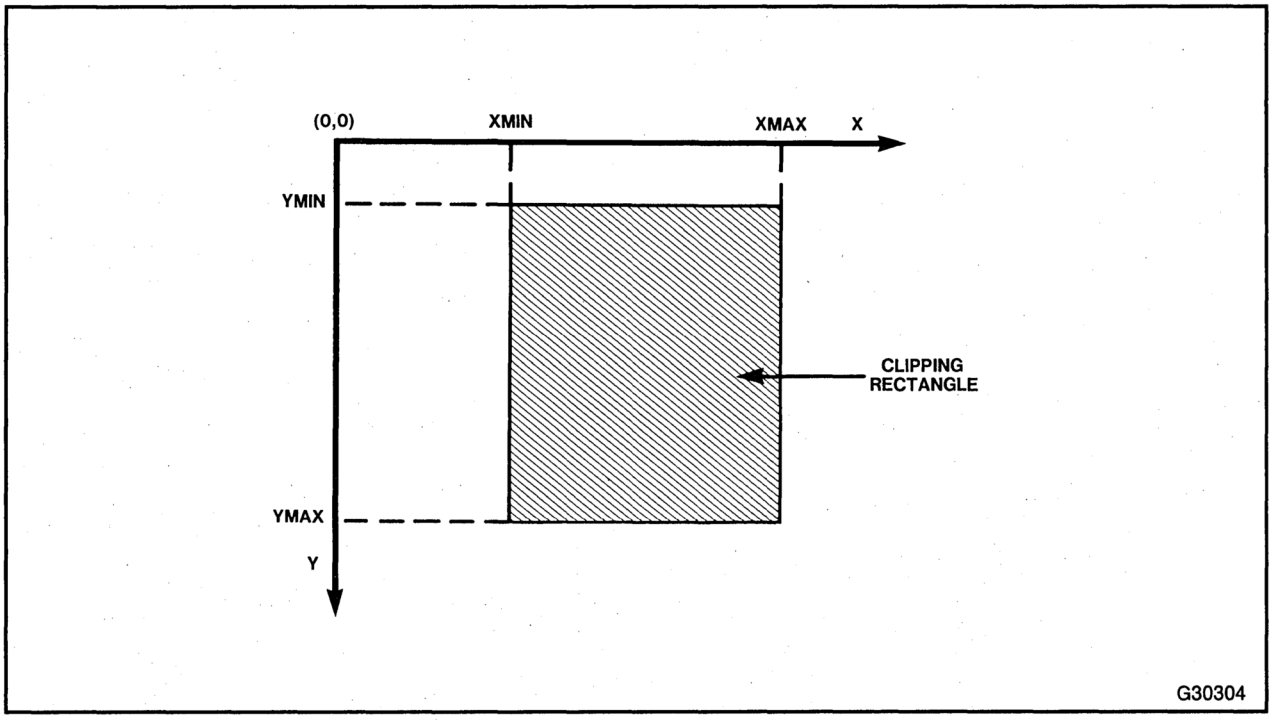


Figure 2-29. Define Clipping Rectangle

2.10.13 DEF_COLORS–Define Colors

Opcode 3DH

Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Opcode 3DH								Reserved							GECL
Foreground Color															
Background Color															

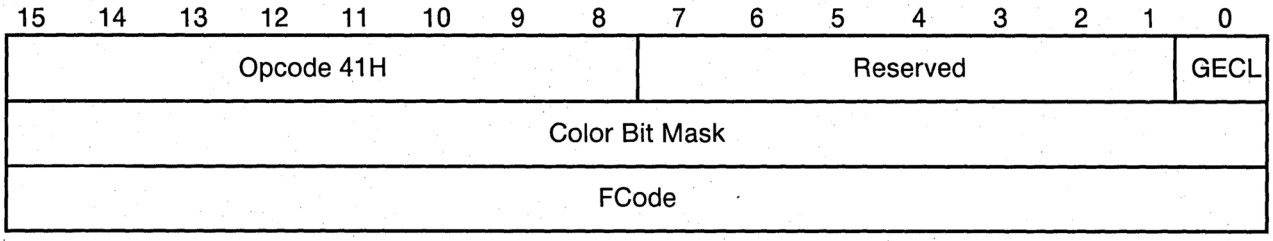
Description The Def_Colors command sets the active foreground and background colors for subsequent drawing operations. Reserved bits should be programmed to zero. Only two colors are active at a time and the two colors remain the same until changed by another Def_Colors command.

The term color refers to the pixel value, which can be 1, 2, 4 or 8 bits wide, depending on the bits per pixel (bpp) defined for the current bitmap. When setting a particular color, the color value must be extended (repeated) across the entire word parameter. For example, in a 4 bpp bitmap, if the required color is 0111, the associated parameter must be set to 0111011101110111.

2.10.14 DEF_LOGICAL_OP–Define Logical Operation

Opcode 41H

Format



Description The Def_Logical_Op command defines the logical operation performed to modify the value of a pixel during any drawing command. Reserved bits should be programmed to zero. Logical operations can combine existing pixel information in a bitmap with new pixel information generated as a result of a bitblt or other drawing operation, such as a bitblt operation which displays only the overlapping regions of two shapes. This example logically combines the contents of two separate bitmap locations to produce new bitmap patterns. When defined, the Def_Logical_Op command applies to all subsequent pixel update operations (line, arc, character, bitblt, etc.) Sixteen binary functions are permitted between the source and destination as shown in Table 2-8.

Table 2-8. Logical Operations

Function	Fcode (Hex)
0	0000
source AND destination	0001
CMP (source) AND destination	0002
Destination	0003
source AND CMP (destination)	0004
source	0005
source XOR destination	0006
source OR destination	0007
CMP (source) AND CMP (destination)	0008
CMP (source) XOR destination	0009
CMP (source)	000A
CMP (source) OR destination	000B
CMP (destination)	000C
source OR CMP (destination)	000D
CMP (source) OR CMP (destination)	000E
1	000F

The Fcode is the truth table of the associated two variable functions. The truth table of the function with the FCode ABCD is diagramed as follows:

Destination	Source 0	Source 1
0	A	B
1	C	D

The following example shows the CMP of source function (FCode = 1010), which replaces the destination with the complement of the source. The truth table shows that if the source equals zero, the destination does not care, the result always will be one. It also shows that if the source is one, the destination does not care, the result always will be zero.

Destination	Source 0	Source 1
0	1	0
1	1	0

The Color Bit Mask parameter in the Def_Logical_Op command preserves the concept of bit planes and their classical roles, while storing pixels in a sequential manner to mask off subsets of the bitmap to all drawing operations such that these subsets of color bit masks represent different information.

The color bit mask has the same number of bits as the bpp of the current bitmap. However, the mask must be extended (repeated) over the entire parameter word. For example, in a 2 bbp bitmap, if the desired pixel mask is 01, then the mask parameter must be 0101010101010101. In the example, a zero (0) in the mask indicates the corresponding color bit is masked and will not be modified during any drawing operation. A one (1) in the mask indicates the corresponding pixel bit may be updated.

The FCode parameter selects the logical function that determines how to assign color values when updating a pixel. Table 2-8 lists the FCode values and corresponding logical functions. Each function is a bit-by-bit logical combination of pixel values. Destination is the current color value of the pixel being updated, which means the existing value of the destination pixel may determine the new color for the pixel.

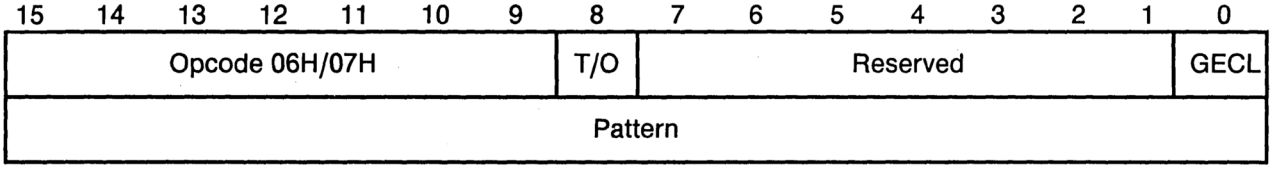
The source value is interpreted according to the type of drawing command performed. For geometric drawing commands, the source value is determined by the current texture, foreground color, and background color (see Def_Colors in Section 2.10.13 and Def_Texture in Section 2.10.15). For character drawing commands, the source is determined by the active character font foreground color and background color. For bitblts, the source is the value of the corresponding pixel in the source rectangle. For transparent textures and transparent character strings, pixels corresponding to zeros in the source pattern are not overwritten irrespective of the FCode.

2.10.15 DEF_TEXTURE–Define Texture

Opcode

06H - Opaque
07H - Transparent

Format



Description

The Def_Texture command sets the active texture for subsequent geometric drawing commands. The 16-bit pattern parameter specifies the texture. Each bit in the pattern corresponds to a pixel, which allows the command to specify a pattern of 16 pixels. Reserved bits should be programmed to zero.

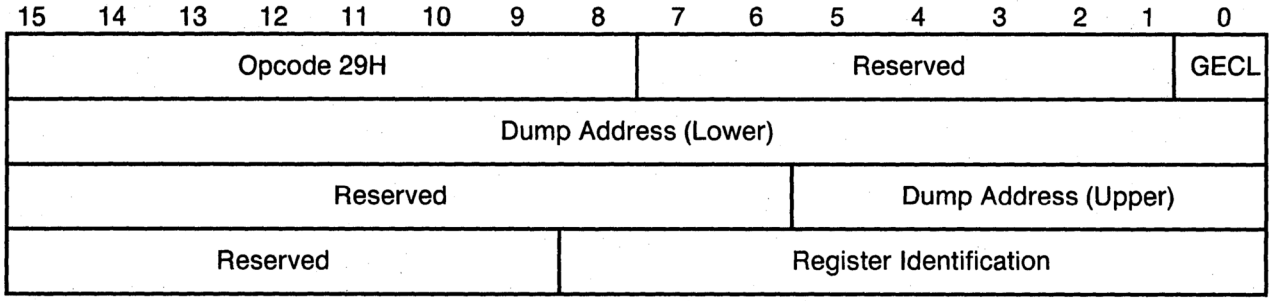
The texture is defined as either opaque or transparent depending on the T/O bit in the opcode. In Transparent Mode, T/O equals one (1). In Transparent Mode, a one in the pattern indicates the corresponding bit should be updated using the foreground color as the source; a zero in the pattern indicates the corresponding bit should not be overwritten. In Opaque Mode, the T/O bit equals zero. In Opaque Mode, a value of one in the pattern signifies that the corresponding pixel is to be written in the foreground color; a zero indicates the corresponding pixel is to be written in the background color.

When a geometric drawing command initializes the pattern, the first pixel drawn corresponds to the most significant bit (MSb) of the defined pattern.

2.10.16 DUMP_REG–Dump Register

Opcode 29H

Format



Description The Dump_Reg command writes the contents of a Graphics Processor (GP) Control or Context Register to the Dump Address specified in the command. To remain compatible with future hardware, access only the registers in Table 2-9. The registers in Table 2-10 should be saved and restored only when switching between multiple tasks that share the GP. The Dump Address must be specified as an even byte address. Reserved bits should be programmed to zero.

A register with a value using 16 bits or less stores its value in a 16-bit word. If the value requires less than 16 bits, the value is stored right justified with the upper unused reserved bits zeroed. If a register has a value requiring more than 16 bits, two consecutive 16-bit words are used. The first word contains the least significant 16 bits. The second word contains the remaining significant bits, which are right justified with the unused, reserved upper bits of the word zeroed. For example, the Stack Pointer (GSP) is stored in two consecutive words. The first word holds the lower address, the least significant 16 bits. The second word holds the higher address; the remaining most significant 6 bits, which are right justified with the upper unused 10 reserved bits zeroed.

Table 2-9. Control Registers

Register Name	Register ID (Number of Bits)	Register Function
GPOEM	0003 (6)	Poll Mask
GIMR	0004 (8)	Interrupt Mask
GSP	010C (22)	Stack Pointer
GCNT	0015 (16)	Character Count

Table 2-10. Context Registers

Register Name	Register ID	Number
REG 5	0007	*(2,2)
REG 6	010B	21
REG 7	010D	21
REG 8	010F	21
REG 9	0010	16
REG 10	0011	16
REG 11	0012	16
REG 12	0013	16
REG 13	0016	16
REG 14	001C	4
REG 16	0090	16
REG 17	0091	16
REG 18	0096	16
REG 19	0097	16
REG 20	0099	16
REG 21	009B	16
REG 22	009C	16

* These bits are right justified in each byte of the word in which they are stored. Two bits are stored in bits 0 and 1 and two bits are stored in 8 and 9; the remaining upper bits in each byte are zeroed.

2.10.17 EBITBLT–Expand Ibpp Source to Currently Defined Bpp

Command	Opcode
BITBLT_EO	D400H
BITBLT_ET	D500H
BITBLT_ERO	D600H
BITBLT_ERT	D700H

Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Opcode						R	T/O	Reserved							GECL
Source Bitmap Origin Address Lower															
Source Bitmap Origin Address Upper															
XMAX (Source Bitmap Width - 1)															
YMAX (Source Bitmap Height - 1)															
Source Rectangle X Coordinate															
Source Rectangle Y Coordinate															
DX (Rectangle Width - 1)															
DY (Rectangle Height - 1)															

Description

Source Rectangle is monochrome and the bpp of Destination Rectangle is same as the last defined in DEF_BITMAP.

COLOR USED IN RASTEROP

R	T/O	SRC PIXEL = 0	SRC PIXEL = 1
0	0	Background	Foreground
1	0	Foreground	Background
0	1	Transparent	Foreground
1	1	Foreground	Transparent

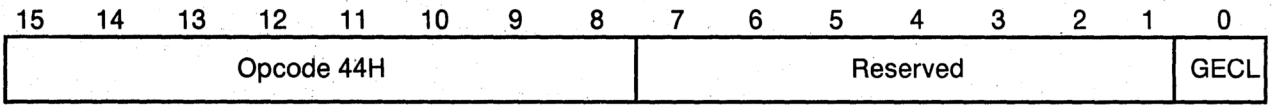
If destination rectangle crosses clip rectangle boundaries, only the portions of the destination rectangle lying within the window are drawn but GBCOV is set. If source rectangle crosses source bitmap boundaries then results are undefined. GBCOV is not necessarily set. Source Bitmap Origin address is even byte. Source Bitmap width needs to correspond to a positive 16-bit word multiple. Allowable values of xmax are 0, 15, 31, ..., 32767. Dx = (width − 1) and dy = (height − 1) of rectangle before clipping. Results are defined only if source rectangle is enclosed within the source window.

GPSC is set if any pixels in destination rectangle lie within the clip rectangle (whether or not they are transparent). GBCOV is set and no bitmap accesses are made if the destination rectangle lies completely outside the clip rectangle. New GCPP = (xc + dx + space, yc) where (xc, yc) = old GCPP (regardless whether clipping occurred or whether in PICK mode).

2.10.18 ENTER_PICK–Enter PICK Mode

Opcode 44H

Format



Description

The Enter_Pick command puts the Graphics Processor (GP) in PICK Mode (see Section 2.6.5). All GP commands execute normally, except pixels are not updated in the bitmap. Instead, the x and y coordinates that are generated are compared to the active clipping rectangle. If a command is executed that would normally update a pixel in the clipping rectangle, the GP sets the Pick Successful Flag (GPSC) in the Status Register (GSTAT). Reserved bits should be programmed to zero.

In PICK Mode, neither Bitmap Overflow Flag (GBCOV or GBMOV) in GSTAT is updated by any command. The Exit_Pick command, described in Section 2.10.18, returns the GP to normal operation. For details on GSTAT, refer to Section 2.2.1.1 "Graphics Processor Status Register (GSTAT)."

2.10.19 EXIT_PICK–Exit PICK Mode

Opcode 45H

Format

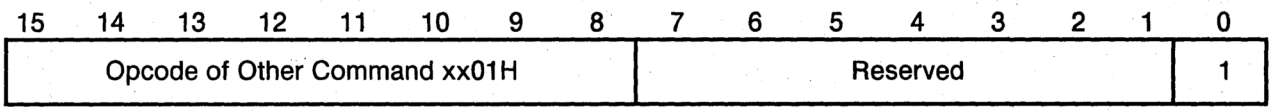
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Opcode 45H								Reserved							GECL

Description The Exit_Pick command exits Pick Mode and returns the Graphics Processor (GP) to normal operation. See the Enter_Pick command in Section 2.10.17 and Section 2.6.5 for details on PICK Mode. Reserved bits should be programmed to zero.

2.10.20 HALT–Stops Command Execution

Opcode xx01H

Format



Description The Halt command stops command execution and forces the Graphics Processor (GP) to enter Poll State. The Halt command can be used with another command specifying a valid opcode, such as Nop, to halt GP command execution. Whatever valid opcode is specified, the command is not executed. Reserved bits should be programmed to zero. For details on Poll State, refer to Section 2.3.2 "Poll State."

2.10.21 INCR_POINT—Draw Series of Incremental Points

Opcode B4H

Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Opcode B4H								Reserved							GECL
Array Base Address (Lower)															
Reserved												Array Base Address (Upper)			
Number of Points															

Description The Incr_Point command draws a series of incremental points into the active bitmap. This command can efficiently draw complex figures. Incremental points are pixels that are adjacent to each other, so the drawn figure resembles a continuous line drawing. The drawing starts at the point indicated by the Graphics Current Position Pointer (GCPP) and continues from point to point in a specified order. When the command completes execution, the GCPP is positioned at the last point specified. If drawing is in transparent mode, the texture must be solid.

Descriptors for incremental points are passed to the Graphics Processor (GP) in an array. The largest allowable single array is 32K points. The command parameters specify the base address of the array. Reserved bits should be programmed to zero. The value n is the number of descriptors; n points are drawn. Because the points are adjacent, only nine possible choices exist for each successive point. Each Incremental Point Descriptor is 4 bits wide. Figure 2-30 shows the array format.

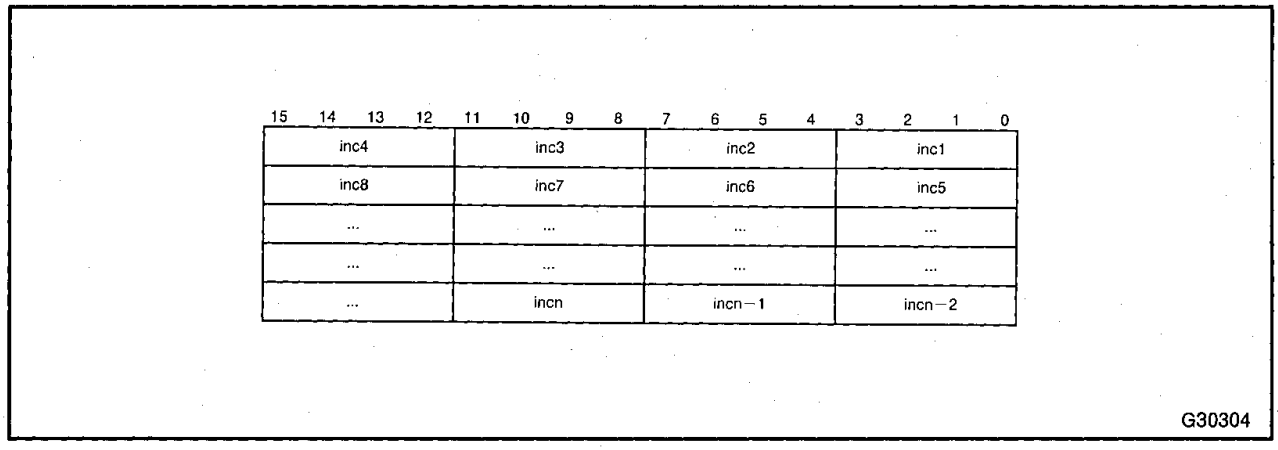


Figure 2-30. Incr_Point Descriptor Array

Drawing starts at the GCPP and continues from point to point as determined by the array. Each 4-bit Descriptor describes the next point to be drawn as shown in Figure 2-31 and Table 2-12. The two high order bits of the nibble select an increment in the x direction. The two low order bits select an increment in the y direction. Table 2-11 illustrates the encoding.

If any specified point lies outside the clipping rectangle (see Section 2.6.4), the Bitmap Overflow Flag (GBMOV) in the Status Register (GSTAT) is set and the point is not drawn. The net result is a clipped figure. The current color, texture, and logical operation are observed.

In PICK Mode (see Section 2.6.5), all pixels (foreground and background) are computed and checked against the clipping rectangle. The bit pattern in memory remains unchanged. If any pixel lies within the clipping rectangle, PICK Mode terminates and the Pick Successful flag (GPSC) in GSTAT is set. For details on GSTAT, refer to Section 2.2.1.1 "Graphics Processor Status Register (GSTAT)."

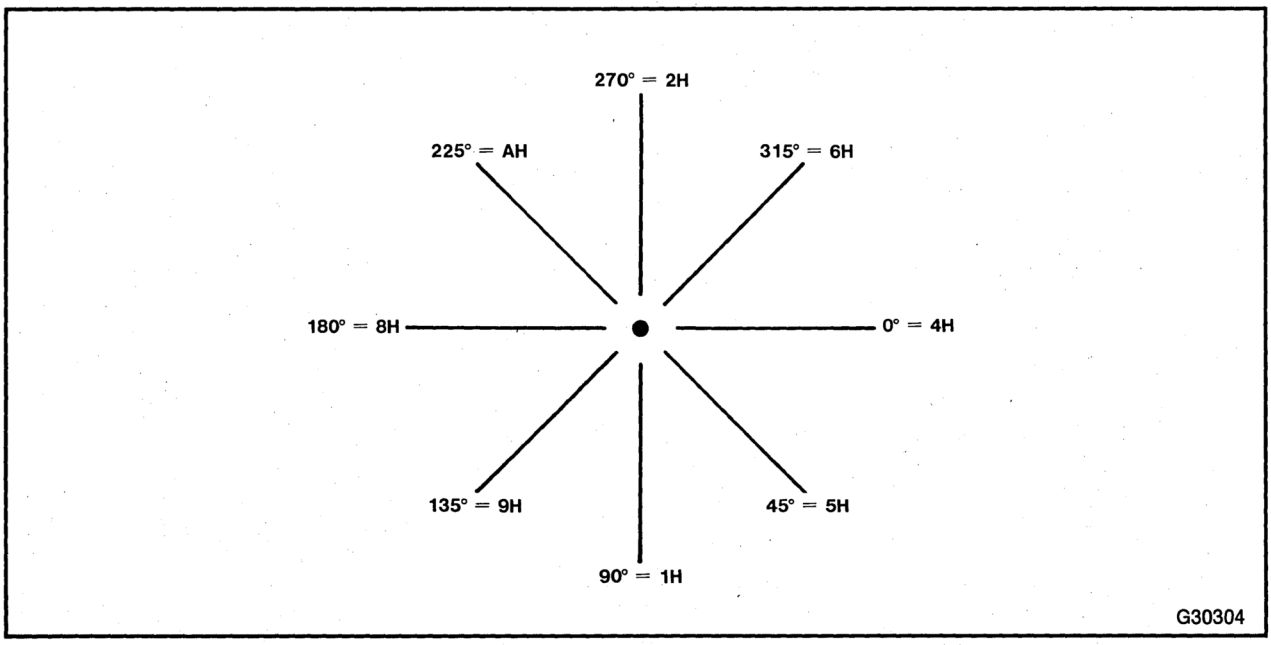


Figure 2-31. Array Values for Incr_Point

Table 2-11. Increment Point Codes

Code	Increment
00	0
01	+1
10	−1
11	illegal

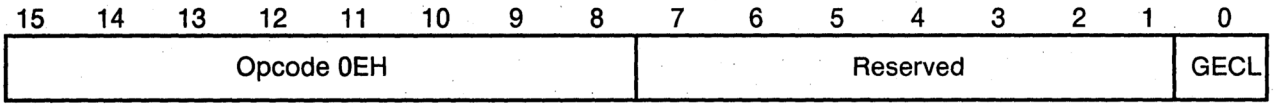
Table 2-12. Incr_Point Command Array Values

Direction (degrees)	x inc	y inc	Code	Value
0	+1	0	0100	4H
45	+1	+1	0101	5H
90	0	+1	0001	1H
135	−1	+1	1001	9H
180	−1	0	1000	8H
225	−1	−1	1010	AH
270	0	−1	0010	2H
315	+1	−1	0110	6H

2.10.22 INTR_GEN–Generate Interrupt

Opcode 0EH

Format



Description The Intr_Gen command can cause the Graphics Processor (GP) to issue an interrupt. The Generate Interrupt Flag (GINT) in the Status Register (GSTAT) is set during execution of this command. Setting the GINT bit can cause an interrupt depending upon the current Interrupt Mask defined in the Graphics Processor Interrupt Mask Control Register (GIMR).

With GSTAT indicating the cause of an interrupt and the BIU Control Register acknowledging the interrupt, both these registers must be read to clear an interrupt.

When using this command, program the reserved bits to zero. For more detail, see Section 2.2.2 "Graphics Processor (GP) Control Registers" and Section 2.3.4 "Exception Handling."

2.10.23 LINE–Draw Line

Opcode

54H - draw endpoint
55H - do not draw endpoint

Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Opcode 54H/55H								Reserved							GECL
dx															
dy															

Description

The Line command draws a line between the Graphics Current Position Pointer (GCPP) and the specified endpoint. The endpoint is specified as a point relative to the GCPP. If the GCPP is defined as (x, y), the new position of the GCPP after this command executes is (x + dx, y + dy). If dx = dy = 0, a point will be drawn at the GCPP only. Reserved bits should be programmed to zero.

For both opcodes, the new GCPP will be at the specified endpoint. The difference between the two is that opcode 54H draws the endpoint while opcode 55H does not draw the endpoint.

The line drawn by this command has the current active texture, color, and logical operation. The pattern is initialized at the beginning of the line (see Figure 2-32); the first pixel drawn corresponds to the most significant bit of the pattern parameter in the Def_Texture command described in Section 2.10.15.

If any point of the line lies outside the clipping rectangle (see Section 2.6.4), the Bitmap Overflow Flag (GBMOV) in the Status Register (GSTAT) is set and the line is clipped.

In PICK Mode (see Section 2.6.5), all pixels (foreground and background) are computed and checked against the clipping rectangle. The bit pattern in memory remains unchanged. If any pixel lies within the clipping rectangle, PICK Mode terminates and the Pick Successful Flag (GPSC) in GSTAT is set.

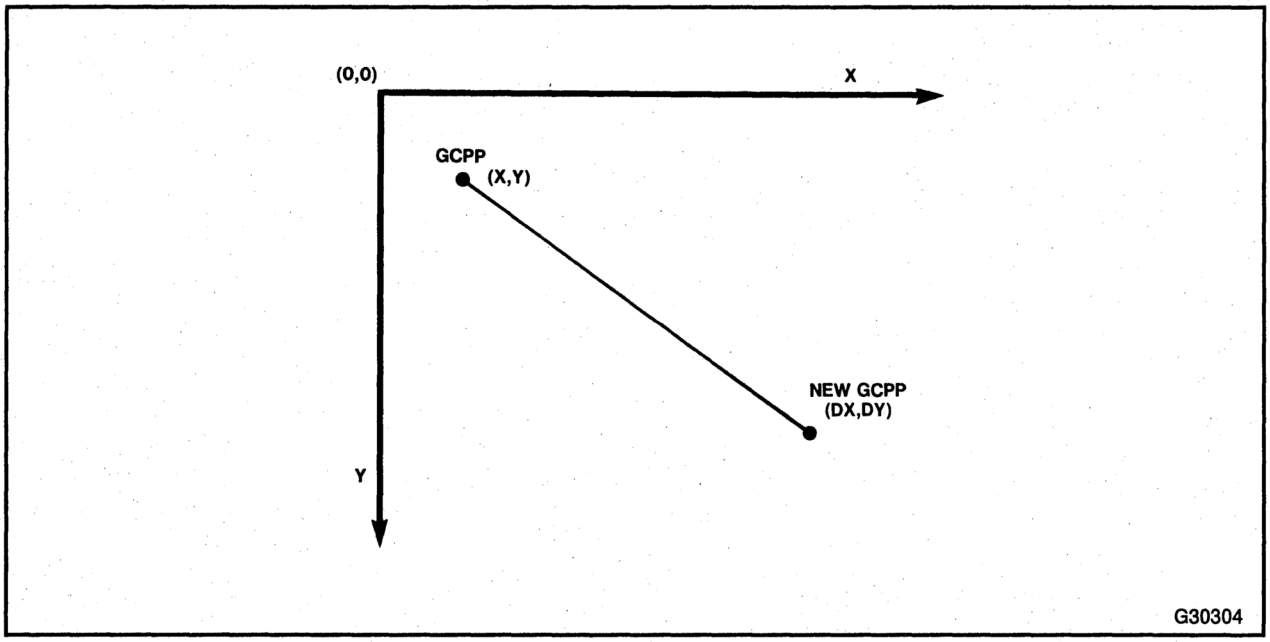
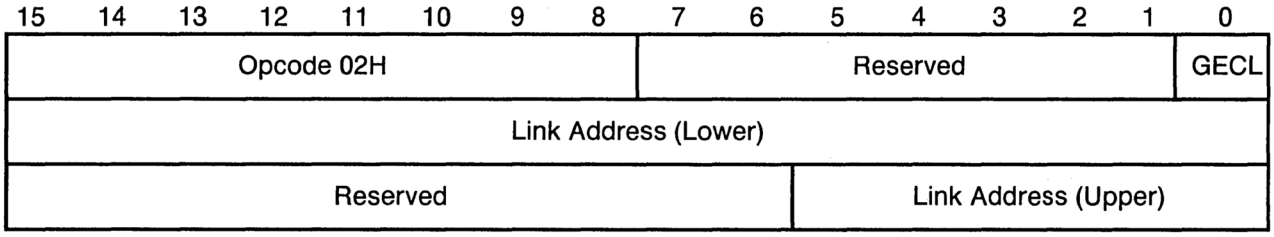


Figure 2-32. Draw Line

2.10.24 LINK—Link Next Command

Opcode 02H

Format

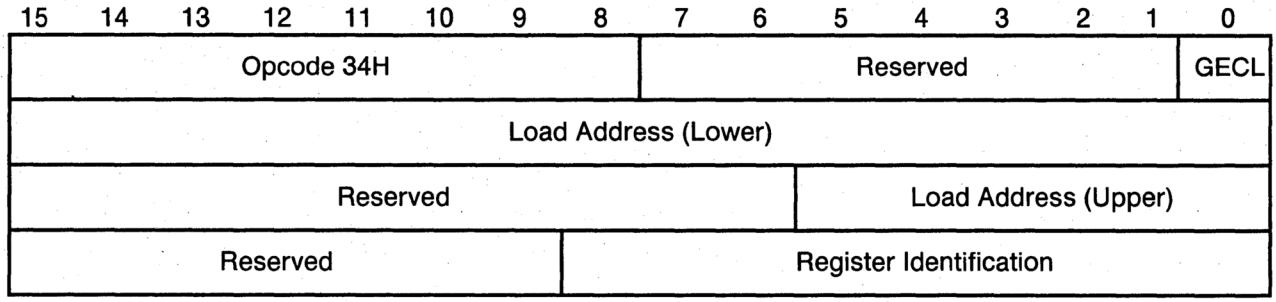


Description The Link command loads the Graphics Processor Instruction Pointer (GCIP) with the specified Link address. No drawing operation is performed. This command is used to branch to subsequent Graphics Command Blocks (GCMBs). The Link Address must be a word (even byte) address. Reserved bits should be programmed to zero. For details on the GCIP, refer to Section 2.2.1.2.

2.10.25 LOAD_REG–Load Register

Opcode 34H

Format



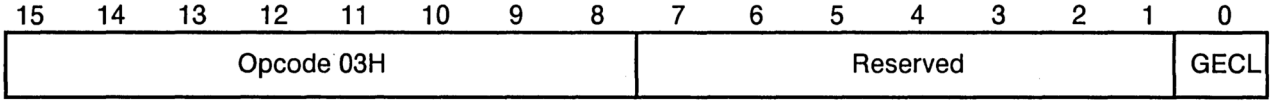
Description The Load_Reg command reads the contents of a specified memory location into a Graphics Processor (GP) Control or Context Register (See Sections 2.2.2 and 2.2.3). The load address must be a word (even byte) address. Reserved bits should be programmed to zero.

Use Load_Reg to load the Stack Pointer (GSP), which is stored in two consecutive words. The lower address holds the least significant 16 bits and the upper address holds the most significant 6 bits which are right justified; the remaining reserved 10 bits must be zeroed.

2.10.26 NOP–No operation

Opcode 03H

Format

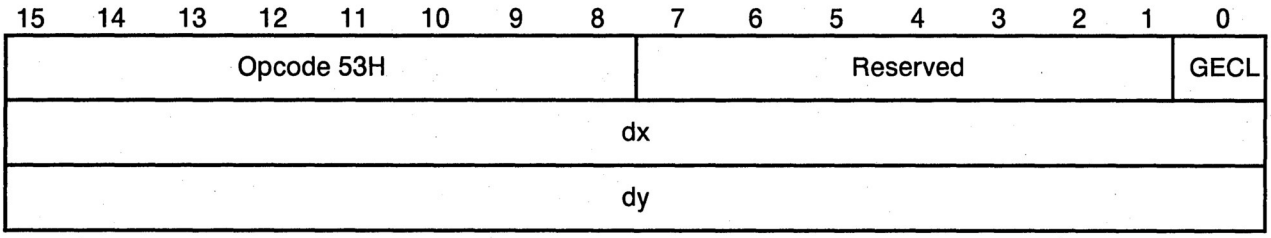


Description The Nop command does not perform any operation. It exists for programming convenience only. After executing the Nop command, the Graphics Processor (GP) proceeds to the next command. The Nop command with the GECL bit set to one is the same as using the Halt command with a valid opcode to stop GP command execution. In both cases, the command is not executed, GP command processing ceases, and the GP enters Poll State. Reserved bits should be programmed to zero.

2.10.27 POINT–Draw Point

Opcode 53H

Format



Description The Point command draws a point at a position relative to the Graphics Current Position Pointer (GCPP). The offsets dx and dy are specified in two's complement form and can be negative. Reserved bits should be programmed to zero. With the GCPP defined as (xc, yc), the new GCPP position following command execution is (xc + dx, yc + dy). The new point uses the currently defined color and logical operation. The point is always drawn in the foreground color.

If the new GCPP lies outside the clipping rectangle (see Section 2.6.4), the point is not drawn and the Bitmap Overflow Flag (GBMOV) in the Status Register (GSTAT) is set. The GCPP moves to the new position as shown in Figure 2-33.

In PICK Mode (see Section 2.6.5), the pixel is computed and checked against the clipping rectangle. The bit pattern in memory remains unchanged. If any pixel lies within the clipping rectangle, PICK Mode terminates and the Pick Successful Flag (GPSC) in GSTAT is set.

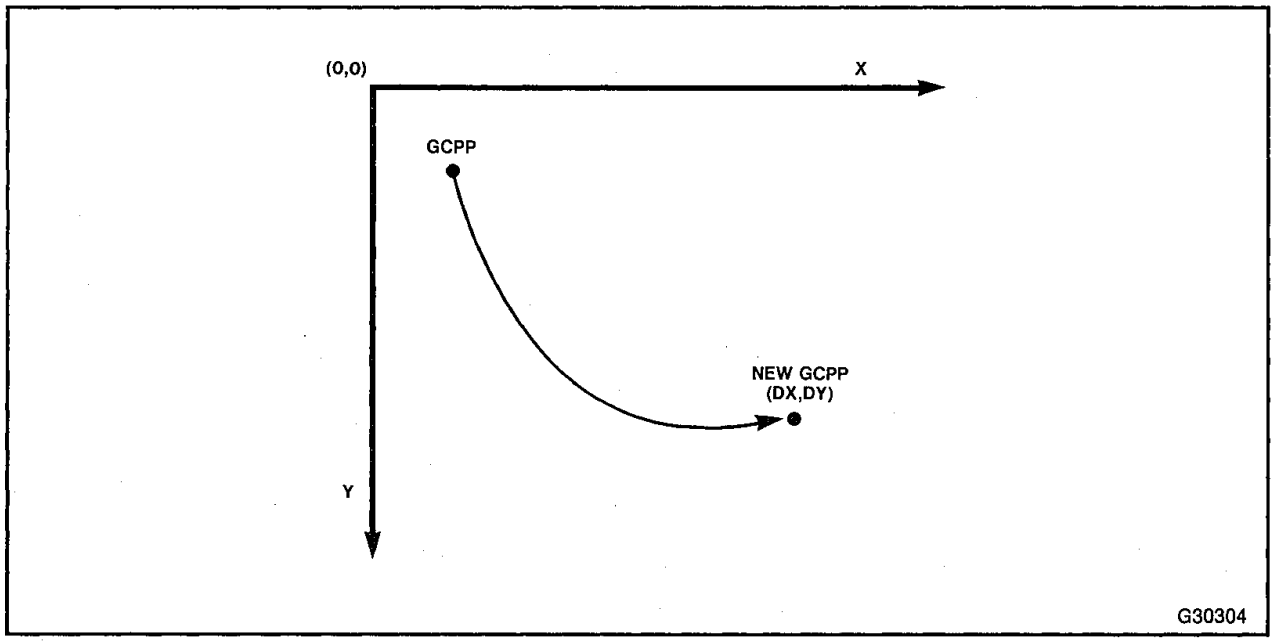


Figure 2-33. Draw Point

2.10.28 POLYGON–Draw Polygon

Opcode 73H

Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Opcode 73H								Reserved							GECL
Array Base Address (Lower)															
Reserved										Array Base Address (Upper)					
Number of Lines															

Description The Polygon command draws a line from the Graphics Current Position Pointer (GCPP) through a series of points specified in an array and then back to the GCPP. The closing line from the nth point to the starting point is drawn automatically. The number of lines actually drawn to complete the polygon is n + 1. The GCPP remains at the initial point. Reserved bits should be programmed to zero.

The lines are drawn in the active texture and color with the currently defined logical operation. The pattern is initialized at the beginning of the first line; the first pixel drawn corresponds to the most significant bit of the pattern parameter in the Def_Texture command, described in Section 2.10.15. The pattern is not initialized for subsequent lines, but continues from one line to the next. The vertices are drawn only once as shown in Figure 2-34.

Information for the vertices is contained in an array, shown in Figure 2-35. The array size should be 2 * N words.

The sides of the polygon proceeds from the Graphics Current Position Pointer (GCPP) and continues to each coordinate in the order specified. If any portion of the line lies outside the clipping rectangle (see Section 2.6.4), the line is drawn clipped and the Bitmap Overflow Flag (GBMOV) in the Status Register (GSTAT) is set.

In PICK Mode (see Section 2.6.5), all pixels (foreground and background) are computed and checked against the clipping rectangle. The bit pattern in memory remains unchanged. If any pixel lies within the clipping rectangle, PICK Mode terminates and the Pick Successful Flag (GPSC) in GSTAT is set.

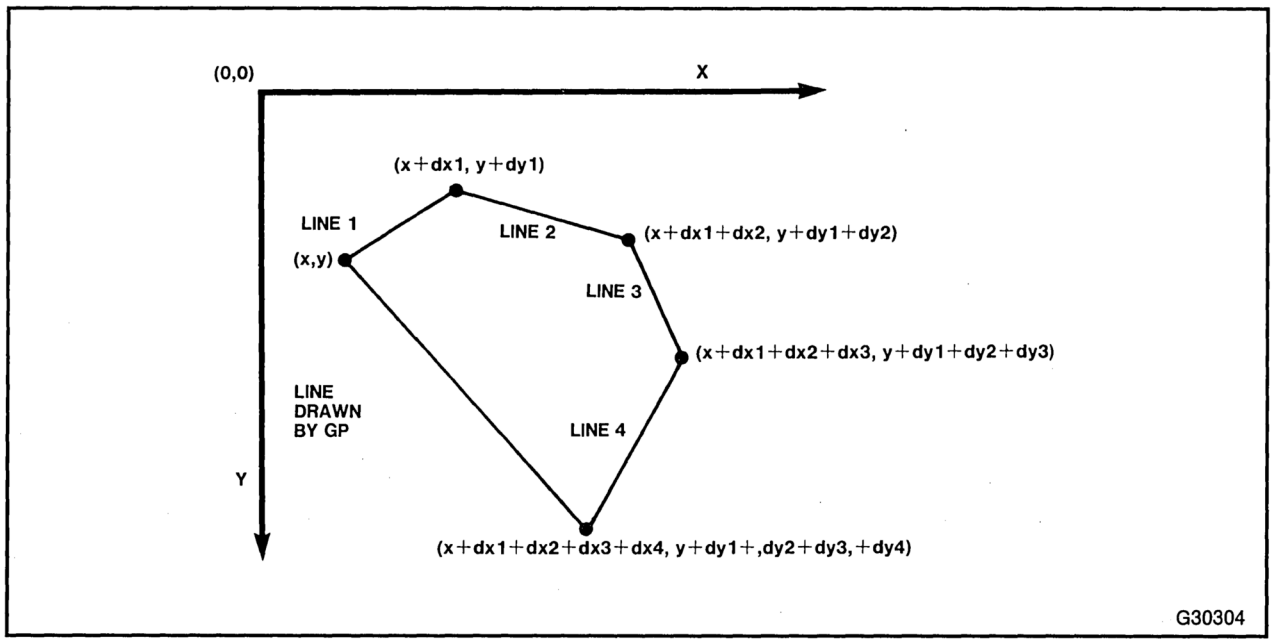


Figure 2-34. Draw Polygon

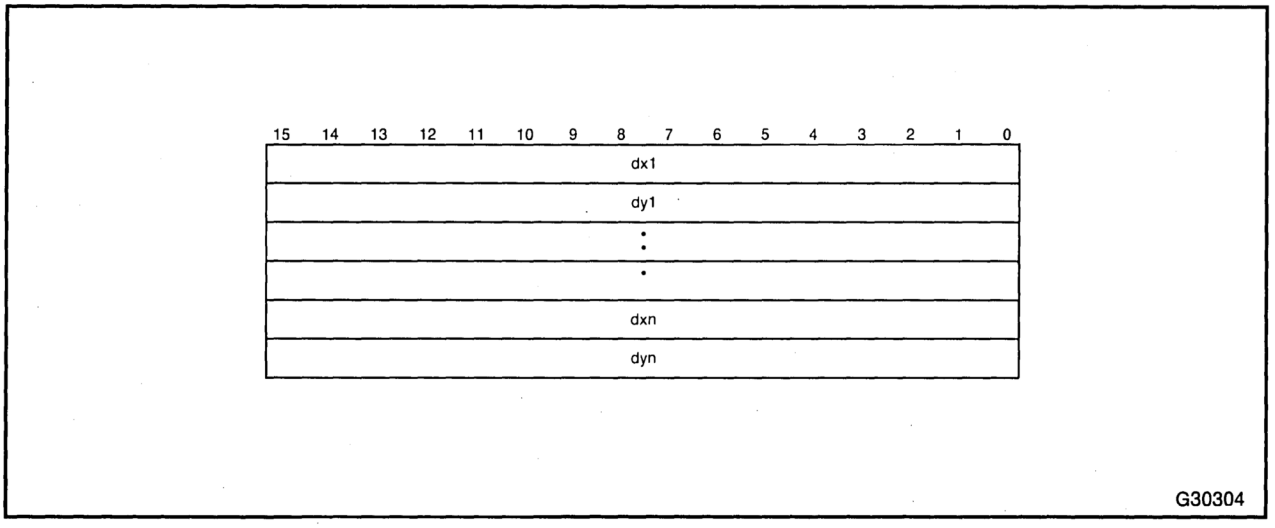


Figure 2-35. Polygon Vertices Array Information

2.10.29 POLYLINE–Draw Polyline

Opcode 74H

Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Opcode 74H								Reserved							GECL
Array Base Address (Lower)															
Reserved										Array Base Address (Upper)					
Number of Lines															

Description The Polyline command draws a line from the Graphics Current Position Pointer (GCPP) through a series of specified points. The lines are drawn in the active texture, color, and logical operation. The pattern is initialized at the beginning of the first line; the first pixel drawn corresponds to the most significant bit of the pattern parameter in the Def_Texture command, described in Section 2.10.15. The pattern is not initialized for subsequent lines, but continues from one line to the next. With an initial GCPP of (x,y), the new GCPP is (x + dx1 + dx2 + ...dxn, y + dy1 + dy2 + ...dyn) as shown in Figure 2-36. Reserved bits should be programmed to zero.

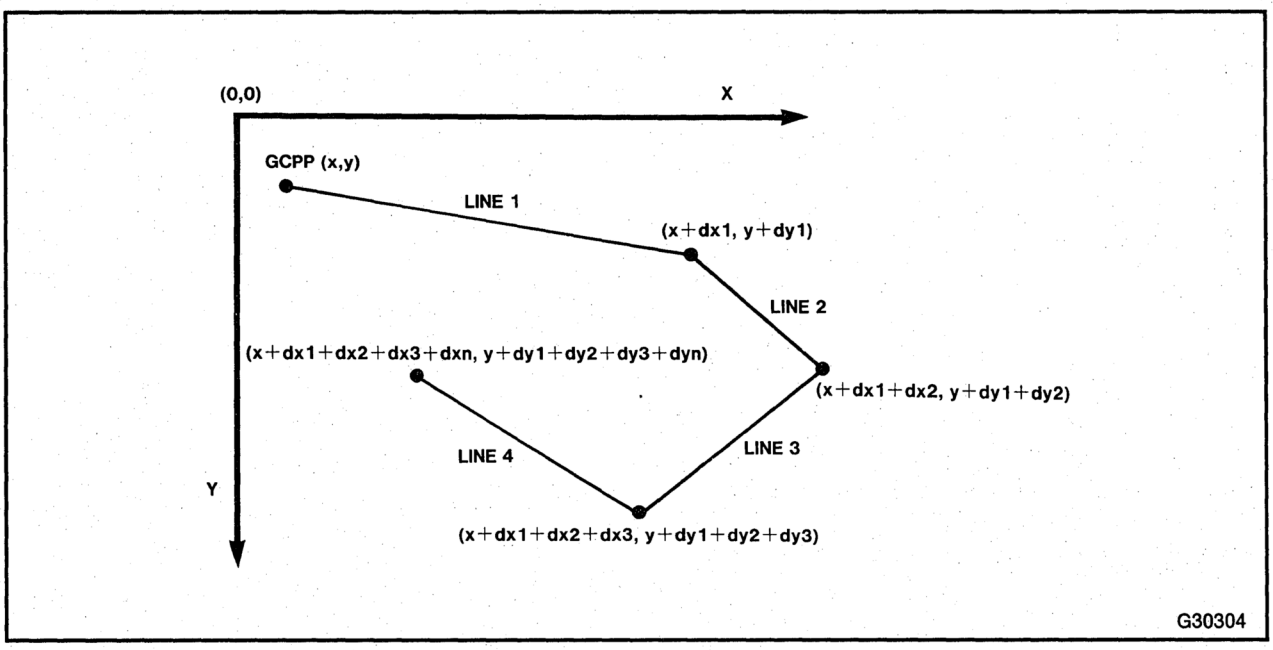


Figure 2-36. Draw Polyline

Information for vertices is contained in an array as shown in Figure 2-37. The size of the array should be 2 * N words.

The polyline proceeds from the GCPP and continues to each coordinate in the order specified. Vertices are not drawn twice. If any portion of the line lies outside the clipping rectangle (see Section 2.6.4), the line is drawn clipped and the Bitmap Overflow Flag (GBMOV) in the Status Register (GSTAT) is set.

In PICK Mode (see Section 2.6.5), all pixels (foreground and background) are computed and checked against the clipping rectangle. The bit pattern in memory remains unchanged. If any pixel lies within the clipping rectangle, PICK Mode terminates and the Pick Successful Flag (GPSC) in GSTAT is set.

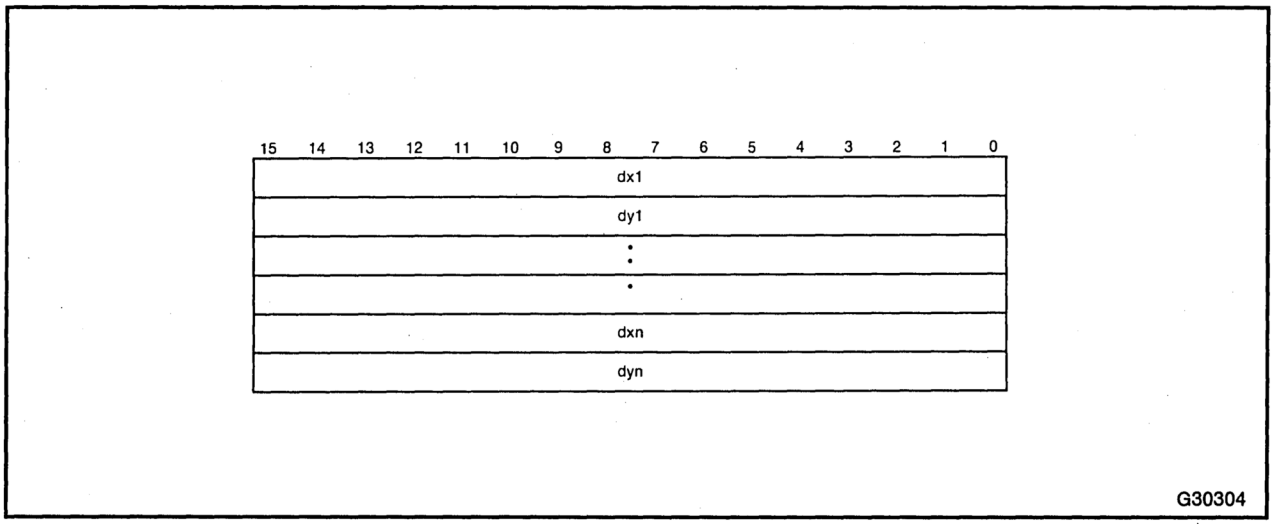


Figure 2-37. Polyline Vertices Array

2.10.30 RECT–Draw Rectangle

Opcode 58H

Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Opcode 58H								Reserved							GECL
dx															
dy															

Description

The Rect command draws a rectangle starting at the Graphics Current Position Pointer (GCPP). Reserved bits should be programmed to zero. The height and width are specified as displacements from the current GCPP and can be negative. Drawing proceeds from the GCPP in the x-direction to the dx value specified. With the GCPP defined as (x,y), the new position of the GCPP after this command executes is (x + dx, y). The new GCPP is at a new x coordinate, but the same y coordinate as shown in Figure 2-38. If dx = dy = 0, one point will be drawn at the GCPP.

The line drawn by this command uses the current active texture, color, and logical operation defined in Def_Logical_Op. The pattern is not initialized for every line in the rectangle but continues from one line to the next adjoining one.

If any point of the line lies outside the clipping rectangle (see Section 2.6.4), the Bitmap Overflow Flag (GBMOV) in the Status Register (GSTAT) is set and the line is clipped.

In PICK Mode (see Section 2.6.5), all pixels (foreground and background) are computed and checked against the clipping rectangle. The bit pattern in memory remains unchanged. If any pixel lies within the clipping rectangle, PICK Mode terminates and the Pick Successful Flag (GPSC) in GSTAT is set.

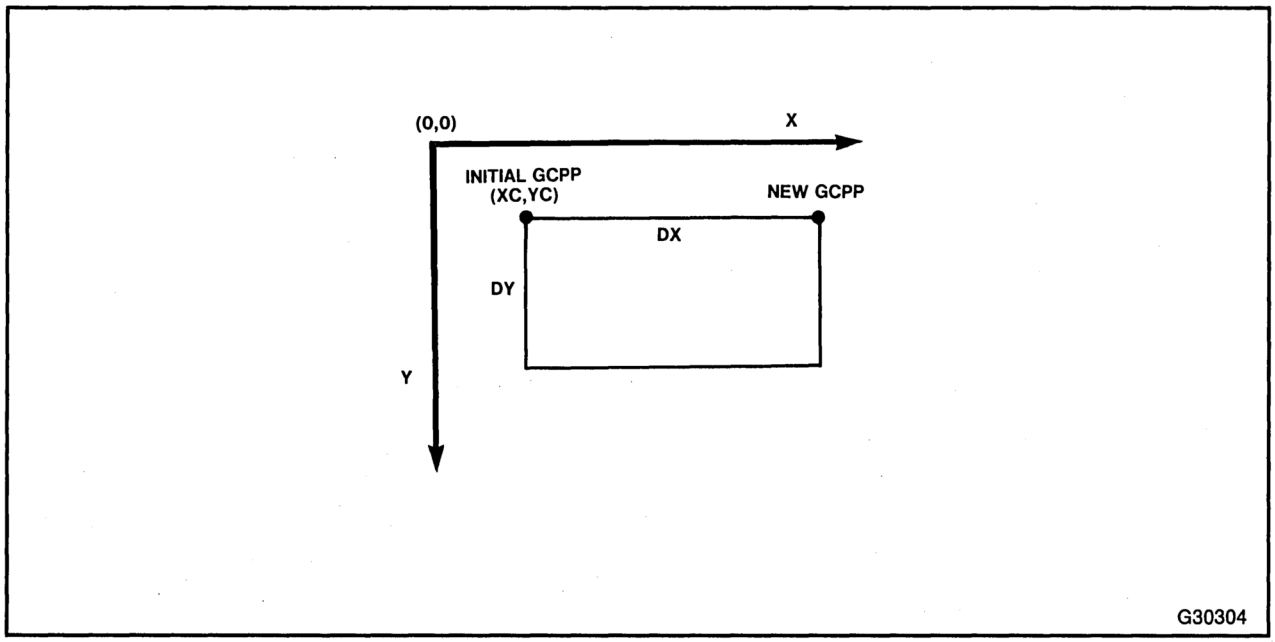


Figure 2-38. Draw Rectangle

2.10.31 REL_MOV—Relative Move

Opcode 52H

Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Opcode 52H								Reserved							GECL
dx															
dy															

Description The Rel_Mov command moves the Graphics Current Position Pointer (GCPP) to a location relative to its current position. Reserved bits should be programmed to zero. The offsets dx and dy are specified in two's complement form and can be negative. With the GCPP defined as (x, y), the new position after this command executes is (x + dx, y + dy) as shown in Figure 2-39.

The Bitmap Overflow Flags (GBMOV and GBCOV) and the Pick Successful Flag (GPSC) are not set as a result of this command.

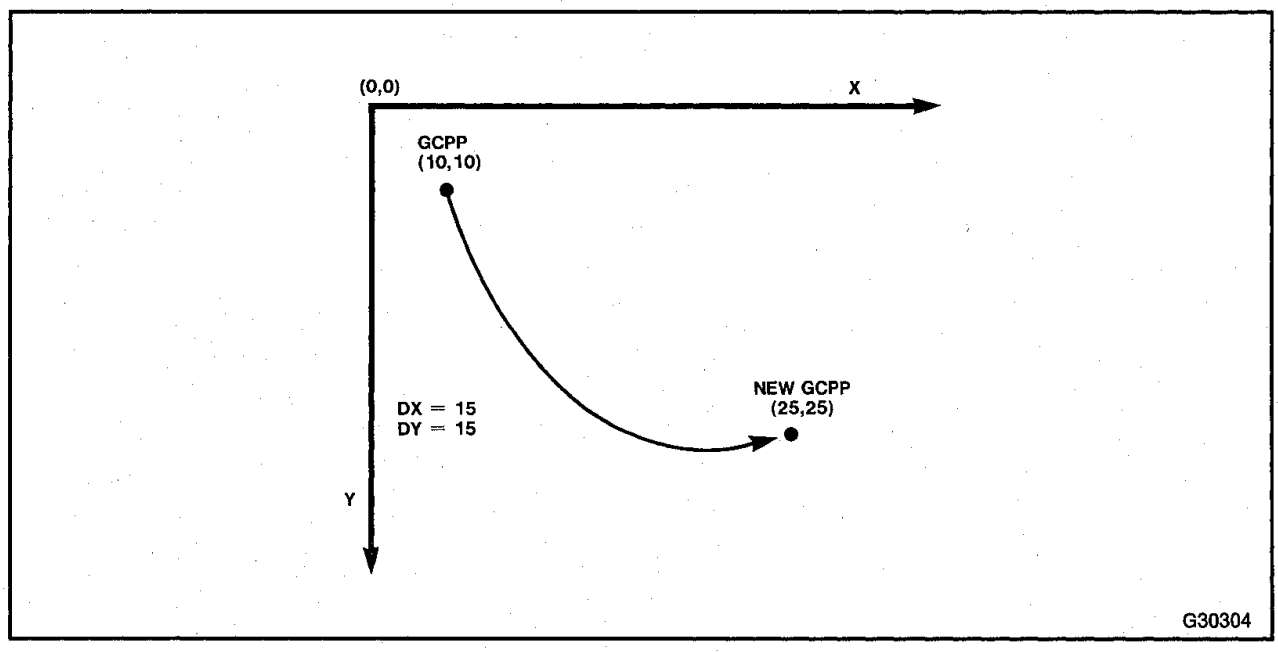
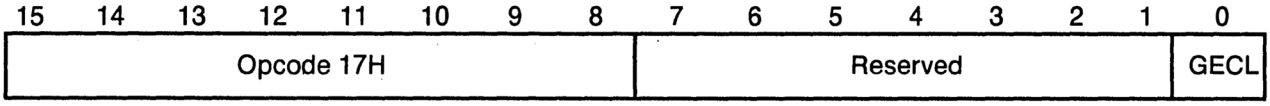


Figure 2-39. Relative Move

2.10.32 RETURN–Return from Subroutine

Opcode 17H

Format



Description The Return command returns command execution from a subroutine to the calling program. Reserved bits should be programmed to zero. See the Call command in Section 2.10.5 for a description of initiating subroutine calls.

When the Graphics Processor (GP) receives a Return command, the GP pops the return address off the stack and transfers it to the Instruction Pointer (GCIP). The Stack Pointer (GSP) is incremented by four. Command execution proceeds with the command addressed by the GCIP, which is the command immediately following the Call command that initiated the subroutine.

2.10.33 SCAN_LINES–Draw Series of Horizontal Lines

Opcode BAH

Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Opcode BAH								Reserved							GECL
Array Base Address (Lower)															
Reserved										Array Base Address (Upper)					
Number of Lines															

Description The Scan_Lines command draws a set of horizontal lines within the specified parameters. This command can fill an area or draw a set of horizontal lines with a defined line pattern.

The lines are drawn with the currently defined logical operation, color, and texture. The pattern is adjusted so that a series of horizontal lines, starting at different x coordinates, appear with their patterns aligned. The texture is aligned to the beginning of the bitmap. Information for the horizontal lines exists in an array as shown in Figure 2-40. The size of the array should be 3 * N words.

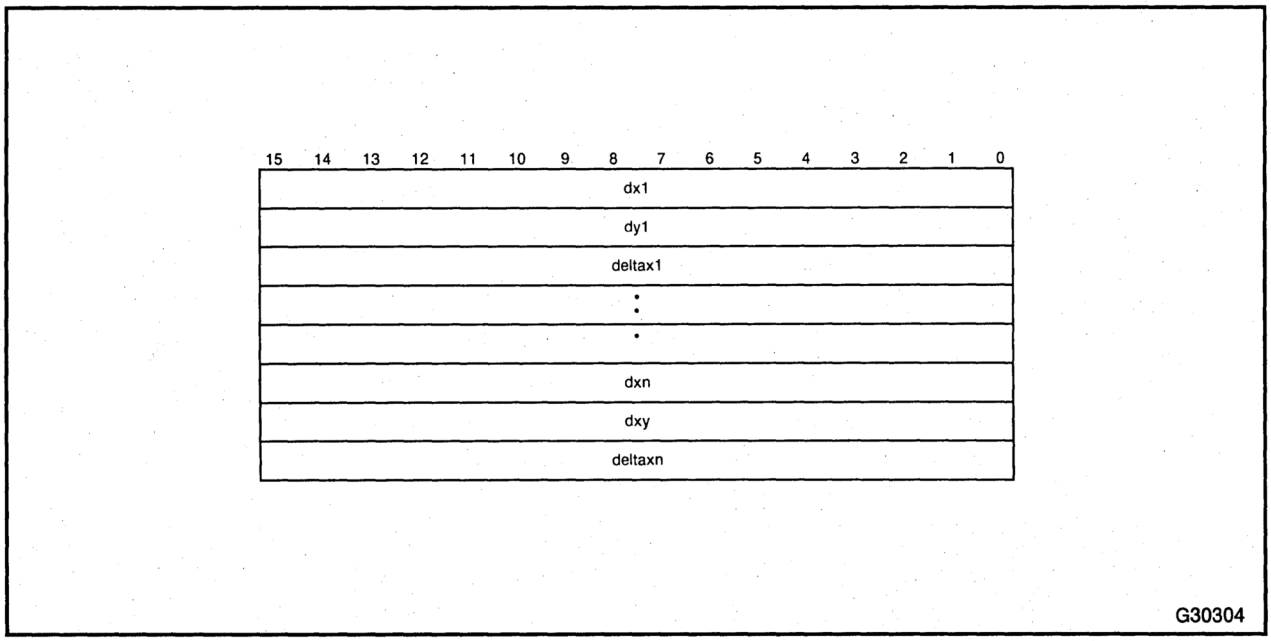


Figure 2-40. Scan_Lines Horizontal Line Array

Reserved bits should be programmed to zero. The arguments dx1 and dy1 specify the relative displacement of the starting point with respect to the Graphics Current Position Pointer (GCPP). The arguments dx and dy move the GCPP to the starting position of the next line. Deltax1 specifies the width of the horizontal line, which can be negative. Deltax has no effect on the next coordinate.

If any portion of the lines lies outside the clipping rectangle, the lines are drawn clipped and the Bitmap Overflow Flag (GBMOV) in the Status Register (GSTAT) is set.

In PICK Mode (see Section 2.6.5), all pixels (foreground and background) are computed and checked against the clipping rectangle. The bit pattern in memory remains unchanged. If any pixel lies within the clipping rectangle, PICK Mode terminates and the Pick Successful Flag (GPSC) in GSTAT is set.

With an initial GCPP of (x, y), after drawing the first line the new GCPP is (x + dx1, y + dy1). When the command completes drawing, the new GCPP is (x + dx1 + dx2 + ...dxn, y + dy1 + dy2 + ...dyn).

Display Processor Overview 3

CHAPTER 3
DISPLAY PROCESSOR OVERVIEW

The Display Processor (DP) is an independent processor within the 82786. The DP is optimized to control the display of video data on a CRT screen, but it also supports other displays. The DP generates Horizontal Synchronization (HSync) and Vertical Synchronization (VSync) timing and Blanking signals, and controls the 8 Video Data (VDATA) output pins. The DP also fetches and loads parameters stored previously in memory.

3.1 DISPLAY PROCESSOR OPERATION

The Display Processor (DP) performs the following functions in generating the display contents for output:

- Generates control and Video Data (VDATA) signals to the display hardware.
- Provides a pointing symbol (cursor).
- Retrieves memory contents of selected bitmaps and outputs corresponding pixels into windows on the display screen.
- Using pixel replication, the DP permits selected portions of bitmaps to be magnified (zooming) horizontally or vertically on the display.
- Loads the shift registers of dual port video random access memories (VRAMs).

Programming the 82786 on-chip Display Control Registers controls the Display Processor. In this way, the application or system software dynamically alters the display content without unacceptable display blinking (flickering) occurring.

The DP uses the memory-mapped DP Internal Registers and register commands to load the Display Control Registers with parameters that have been set up in memory by the CPU. The CPU can update these parameters at any time during screen refresh. The DP automatically synchronizes loading with Vertical Blanking (VBlank). For details on the DP register commands, see Section 3.6. For Display Control Registers, see Section 3.3.2. For the Internal Registers, see Section 3.3.1.

3.1.1 Bitmap Organization

The Display Processor (DP) is optimized to display data in sequential bitmap form. Usually, the Graphics Processor (GP) writes the bitmap in memory, but the host CPU also can be used. The DP can display 1, 2, 4, or 8 bits per pixel (bpp). Bits for each pixel are stored sequentially in bitmap form. The first left-hand pixel to be displayed occupies the most significant bit (MSb) of a word in memory; subsequent pixels occupy sequentially lower bits in the word. The number of pixels per word varies based on the bpp (see Section 3.2 "Video Interface"). Word addresses, moving left across the screen and from the top downward, occupy ascending word address locations within the bitmap.

3.1.2 IBM PC Bitmap Format Support

In addition to the 82786 native bitmap format, the Display Processor (DP) supports the swapped byte IBM Personal Computer (PC), Color Graphics Adapter (CGA) and PCJr bitmap formats. Differing from the 82786 format, the IBM PC format positions the least significant byte of a word to the left of the most significant byte on the screen; whereas, the 82786 format positions the least significant byte to the right of the most significant byte as shown in Figure 3-1.

The DP also supports the 2-bank CGA (see Figure 3-2) and 4-bank PCJr (see Figure 3-3) bitmap formats used by the IBM PC CGA and PCJr systems. This enables bitmaps created by an IBM Personal Computer, PCJr, or compatible system to be upward compatible with 82786 displays. The IBM PC format bitmaps are displayed either as the whole screen or as windows on a screen with 82786 created bitmaps. The IBM PC bitmaps can be zoomed, or used with interlaced or accelerated displays. The 2-bank CGA and 4-bank PCJr modes are not supported in interlace mode.

Figure 3-2 illustrates a 2 bank, swapped byte memory bitmap for a CGA 4-color medium resolution display 80×200.

Figure 3-3 illustrates a swapped byte, 4 bank memory bitmap for the PCJr with a 16-color medium resolution (4 bpp) and 4-color high resolution (2 bpp) display 160×200.

Although the DP can display bitmaps created in these formats, the Graphics Processor (GP) always draws bitmaps in 82786 format. The vertical mapping of IBM PC format bitmaps is restricted in that the memory start address of an IBM PC format window must be in the first of the 2 or 4 banks. For more details, see the PC parameter in Table 3-1 "Tile Descriptor Parameters" and Figure 3-6 "Strip and Tile Descriptors" in Section 3.1.3.2 "Strip Descriptor Format."

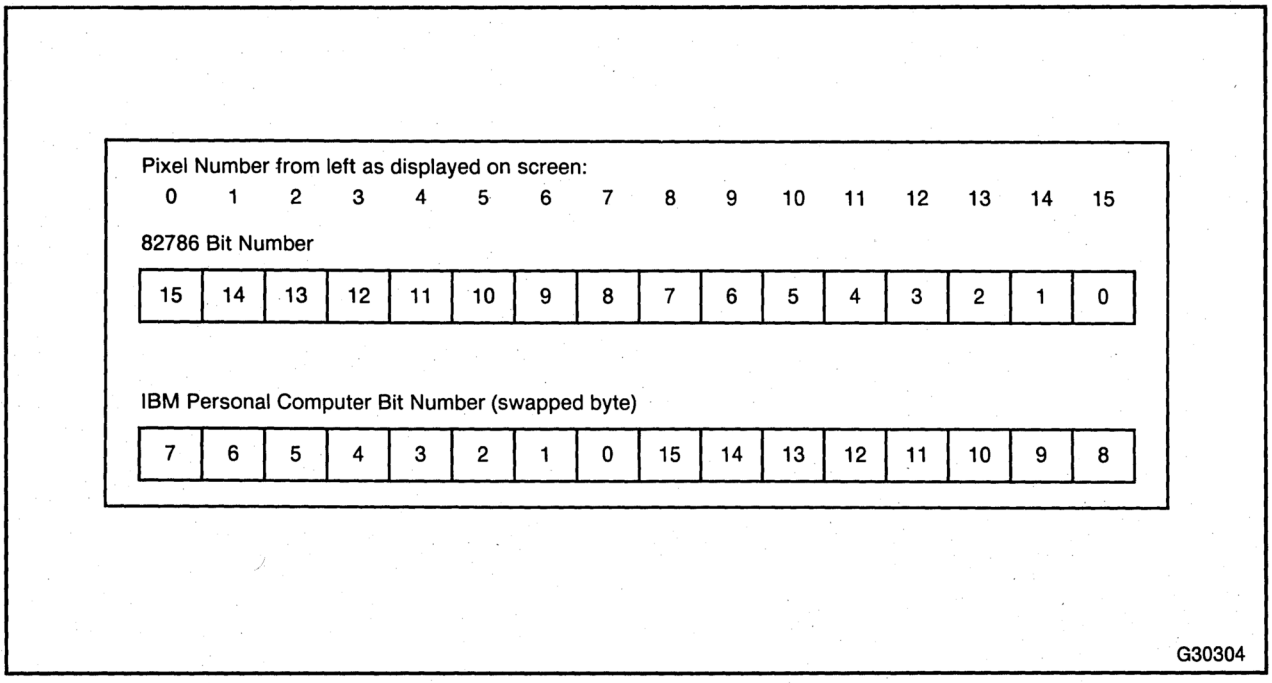


Figure 3-1. 82786 and IBM PC Swapped Byte Bitmap Formats

0	1	2	3	4	5	6	——	4D	4E	4F
2000	2001	2002	2003	2004	2005	2006	——	204D	204E	204F
50	51	52	53	54	55	56	——	9D	9E	9F
2050	2051	2052	2053	2054	2055	2056	——	209D	209E	209F
A0	A1	A2	A3	A4	A5	A6	——	ED	EE	EF
20A0	20A1	20A2	20A3	20A4	20A5	20A6	——	20ED	20EE	20EF
F0	F1	F2	F3	F4	F5	F6	——	13D	13E	13F
20F0	20F1	20F2	20F3	20F4	20F5	20F6	——	213D	213E	213F
140	141	142	143	144	145	146	——	18D	18E	18F
2140	2141	2142	2143	2144	2145	2146	——	218D	218E	218F
:	:	:	:	:	:	:	——	:	:	:
:	:	:	:	:	:	:	——	:	:	:
:	:	:	:	:	:	:	——	:	:	:
1EA0	1EA1	1EA2	1EA3	1EA4	1EA5	1EA6	——	1EED	1EEE	1EEF
3EA0	3EA1	3EA2	3EA3	3EA4	3EA5	3EA6	——	3EED	3EEE	3EEF
1EF0	1EF1	1EF2	1EF3	1EF4	1EF5	1EF6	——	1F3D	1F3E	1F3F
3EF0	3EF1	3EF2	3EF3	3EF4	3EF5	3EF6	——	3F3D	3F3E	3F3F

G30304

Figure 3-2. IBM PC CGA Swapped Byte, 2 Bank Example

0	1	2	3	4	5	6	——	9D	9E	9F
2000	2001	2002	2003	2004	2005	2006	——	209D	209E	209F
4000	4001	4002	4003	4004	4005	4006	——	409D	409E	409F
6000	6001	6002	6003	6004	6005	6006	——	609D	609E	609F
A0	A1	A2	A3	A4	A5	A6	——	13D	13E	13F
20A0	20A1	20A2	20A3	20A4	20A5	20A6	——	213D	213E	213F
40A0	40A1	40A2	40A3	40A4	40A5	40A6	——	413D	413E	413F
60A0	60A1	60A2	60A3	60A4	60A5	60A6	——	613D	613E	613F
:	:	:	:	:	:	:	——	:	:	:
:	:	:	:	:	:	:	——	:	:	:
:	:	:	:	:	:	:	——	:	:	:
1EA0	1EA1	1EA2	1EA3	1EA4	1EA5	1EA6	——	1F3D	1F3E	1F3F
3EA0	3EA1	3EA2	3EA3	3EA4	3EA5	3EA6	——	3F3D	3F3E	3F3F
5EA0	5EA1	5EA2	5EA3	5EA4	5EA5	5EA6	——	5F3D	5F3E	5F3F
7EA0	7EA1	7EA2	7EA3	7EA4	7EA5	7EA6	——	7F3D	7F3E	7F3F

G30304

Figure 3-3. Swapped Byte, 4 Bank IBM PCJr Example

3.1.3 Window Display Format

Windows can be displayed on the screen in flexible formats. Windows are divided into segments called tiles. The portion of a window on a scan line is a tile. Up to 16 tiles can be displayed on a horizontal display line and an unlimited number of tiles can be displayed vertically.

At video rates up to 25 MHz and with 8 bits per pixel (bpp), windows can be placed at pixel resolution on the screen, and mapped at pixel resolution in the bitmap. A border can be displayed on any or all sides of each tile on the inside edge of the tile. The BdrColor Register in the Display Control Block defines the border color (see Section 3.3.2 "Display Control Block Registers"). With color centrally defined in the Display Control Block, rather than the Tile Descriptor (see Figure 3-6), all borders for all windows are always the same color. The border can be turned off or on for individual tiles (see the TBLR parameter in Table 3-1 of Section 3.1.3.2 "Strip Descriptor Format"). The border width is always one pixel; it does not vary with the pixel resolution of Accelerated Modes, which can be 1, 2, 4, or 8 depending on the mode (see Section 3.2.1 "CRT Controller") and dot rate (see Section 3.2.2 "Video Rates").

A single, programmable, background color can be displayed in areas not covered by tiles. Use of background color minimizes system bandwidth because data is fetched for tiles only. Significant DP bandwidth reductions can be realized for many applications, which can increase bandwidth availability for the CPU and Graphics Processor (GP). For details on setting the background color, refer to the F bit in Table 3-1 in Section 3.1.3.2. "Strip Descriptor Format" and the FldColor Register in Table 3-8 of Section 3.3.2 "Display Control Block Registers."

3.1.3.1 STRIP DESCRIPTORS

The screen area with its windows (which can be overlapping) is divided into strips of arbitrary width and height as shown in Figure 3-4.

The horizontal format of tiles remains constant for an entire strip, which creates a rectangular area that is easy to manage. Circular or irregular shaped windows can be created by defining a new strip every display line as illustrated in Figure 3-5.

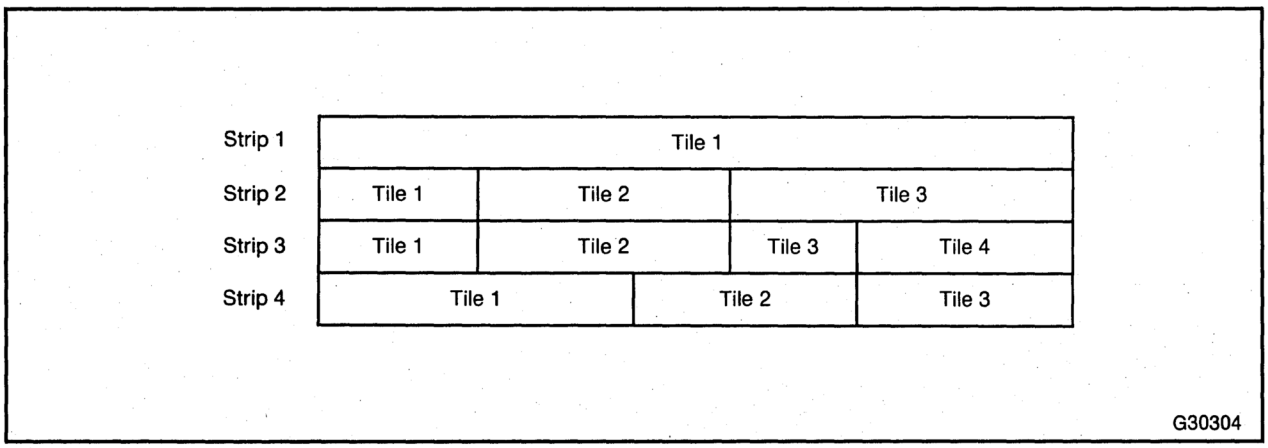


Figure 3-4. Screen Composition with Strips and Tiles

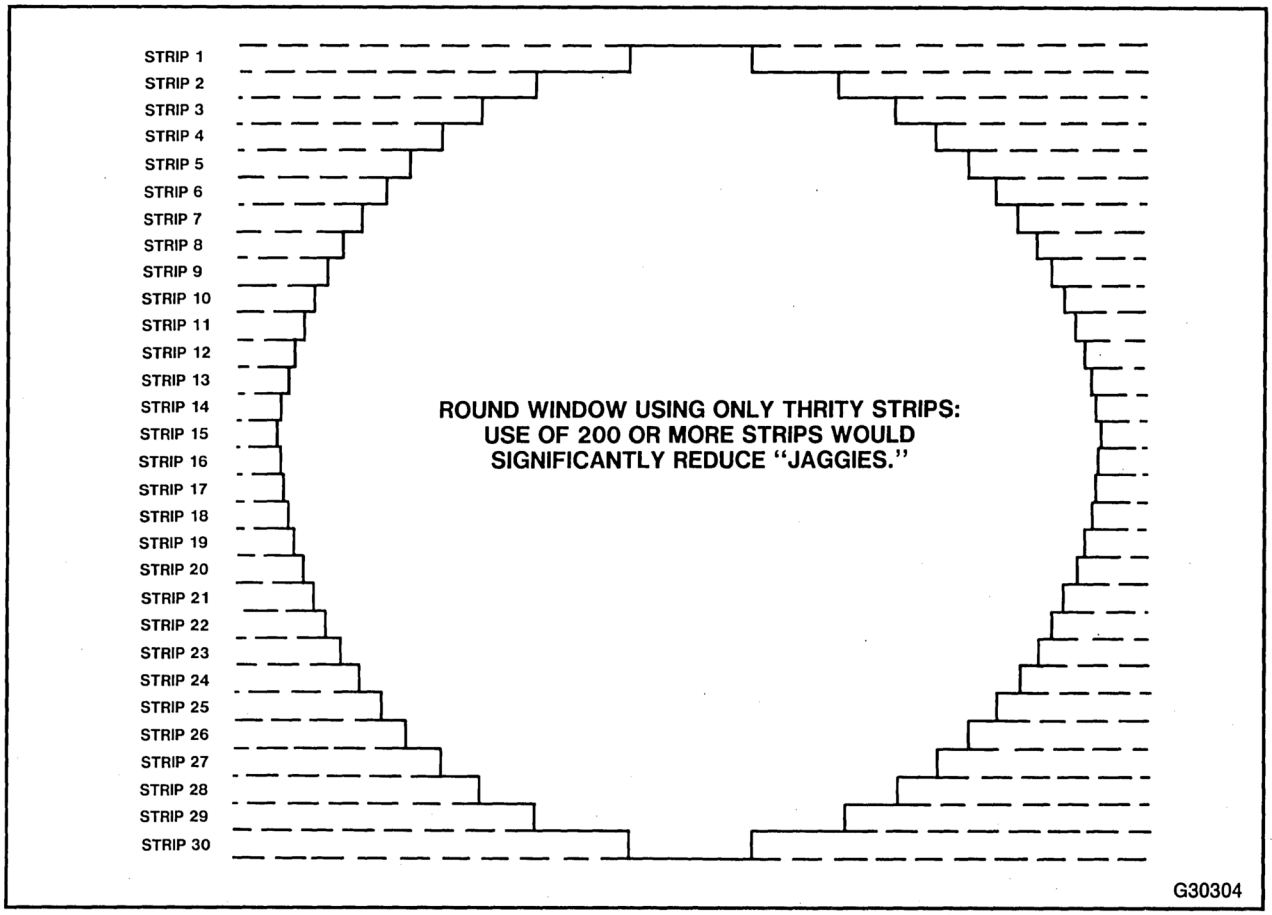


Figure 3-5. Irregular Shaped Window

The CPU creates memory-resident Strip and Tile Descriptors defining the number of tiles in each strip, the memory start address, fetch count, and bpp. The CPU updates these memory-resident Strip Descriptors only when the window arrangement on the screen changes. For details, see Section 3.1.3.2 "Strip Descriptor Format."

Before each strip is displayed, the Display Processor (DP) reads the Strip Descriptor for the strip and stores the information in its internal memory. The DP uses this information to do all subsequent calculations required to display the strip.

3.1.3.2 STRIP DESCRIPTOR FORMAT

The CPU sets up the Strip Descriptors in memory for the Display Processor (DP). A Strip Descriptor exists for each strip of window tiles to describe the tiles within the strip. Each Strip Descriptor consists of a Header followed by one or more Tile Descriptors. The information is ordered left to right as the tiles appear on the display screen. The Strip Descriptor for a particular strip and all associated Tile Descriptors must be contiguous in memory. A linked list of Strip Descriptors can be created by writing the address of the subsequent Strip Descriptor in the Link to Next Strip Descriptor parameter in the header (see Figure 3-6) of each Strip Descriptor in a series of Strip Descriptors. The addresses must link Strip Descriptor strips together from top to bottom to reflect the order in which the strips appear on the

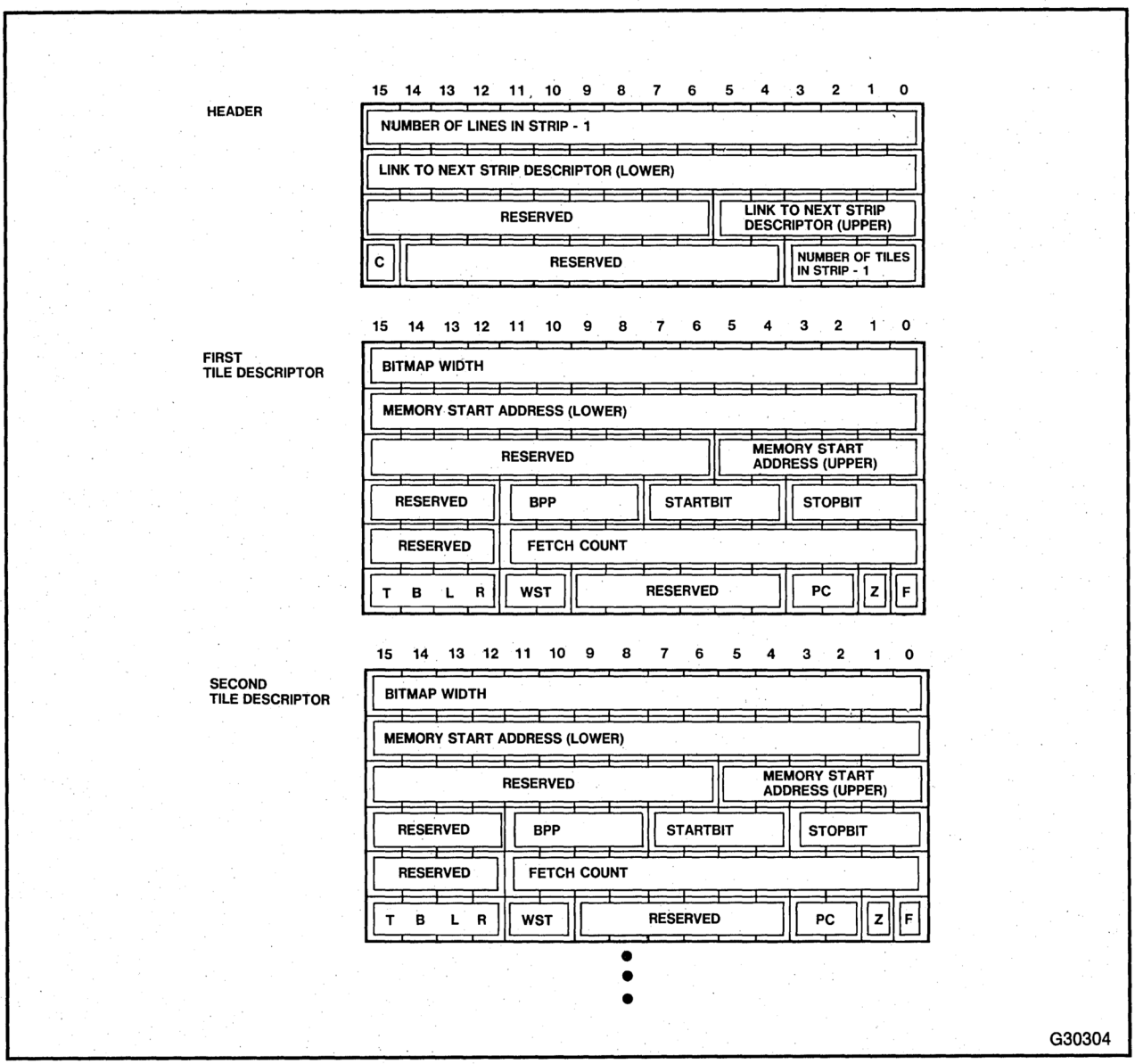


Figure 3-6. Strip and Tile Descriptors

display screen. The Strip Descriptor for the first strip is accessed during the Vertical Blanking (VBlank) Interval by using the address specified in the Descriptor Address Pointer Registers (see Table 3-8 in Section 3.3.2 "Display Control Block Registers"). Using the Link to Next Strip Descriptor parameter in the Header of the first Strip Descriptor, the DP accesses subsequent Strip Descriptors for the frame after completing the display data fetch for the last line of the previous strip.

You must define in the strip descriptors no more scan lines than will actually be displayed. For VRAM systems, the first strip must be at least two scan lines (see section 3.2.5).

The C bit, the most significant bit in the Number of Tiles in Strip parameter in the header also can indicate the default background color when the display area is greater than the number of defined Strip Descriptors as shown in Figure 3-7. For example, if two strips are

defined, but do not fill the display area, set the C bit in the Number of Tiles in Strip parameter of the second strip to one (1) and the DP automatically displays the background color defined in the FldColor Register for the remaining display area (see Table 3-8 in Section 3.3.2 "Display Control Block Registers").

After the first Strip Descriptor is read, subsequent Strip Descriptors are read into the Display Processor (DP) as a single block read after fetching the last display data of the last line of the current strip. The worst case scenario occurs during the Horizontal Blanking (HBlank) time. With the use of interleaved Fast Page Mode DRAMs, the maximum data transfer rate is 50 ns per word. A maximum length Strip Descriptor containing a 4-word header and 16 6-word Tile Descriptors contains 100 words of data. To read the data, 50 cycles, or 5 μS at 10 MHz are required, excluding the few cycles of overhead to get onto the bus. Most CRTs have HBlank times well in excess of 5 μS, but a few may have problems reading in all 16 Strip Descriptors. In such systems, the 82786 may be restricted to using a lower number of windows horizontally across the screen.

If the Strip Descriptor list defines a window that extends beyond the active display area, the Display Processor (DP) displays only the upper left-hand portion of the window and truncates the remaining window. If the Strip Descriptor defines the display portion(s) to be smaller than the active display area and the C bit in the Tile Descriptor Number of Tiles in Strip parameter (see Table 3-1) is set, the remaining active display area is padded with the background color defined in the FldColor Register (see Table 3-8 in Section 3.3.2 "Display Control Block").

The Strip Descriptor header is followed by a Tile Descriptor for each tile the window contains. Table 3-1 outlines the parameters in each Tile Descriptor.

NOTE

The first tile of any scan line must be greater than 1 pixel.

Table 3-1. Tile Descriptor Parameters

Bitmap Width	The 16-bit width of the bitmap value defines the bitmap width in bytes. It must be even because the Display Processor and Graphics Processor use 16-bit word addresses. The bitmap width is added to the memory address for each scan line in the window (the Horizontal Synchronization (HSync) period within the strip) to get the starting address of the next display line (if y-zoom is inactive or has already been calculated). For interlaced displays, the Memory Address is incremented by twice the bitmap width.
Memory Start Address	The 22-bit memory address contains the address of the first word of bitmap data for the window tile. This is an even byte address, which is always the first word in the top left corner of the bitmap.
Bpp	The 4-bit Bpp in bits 11:8 defines the number of bits per pixel in the current window tile. The Bpp can be programmed to 1, 2, 4, or 8 when DRAMs are used at 25 MHz. With VRAMs, the Bpp must be set to zero for the first Tile Descriptor. If Field bit is set to one; these bits are used with the StartBit and StopBit parameters to denote the (number of pixels −1) of background color to display. Program this 12-bit field to number of pixels −1.

Table 3-1. Tile Descriptor Parameters (Cont'd.)

StartBit	The 4-bit StartBit in bits 7:4 defines the bit position in the first memory word that corresponds to the first bit of the first pixel in the tile. The StartBit gives bit resolution to the Memory Start Address and pixel resolution to the start of the window. When DRAMs are used, the programmed Bpp must be consistent with the bpp defined for Bpp in the Tile Descriptor. With VRAMs, the StartBit must be set to zero for the first Tile Descriptor. If the Field bit (F) is set to one, these bits are used with the Bpp and StopBit parameters to denote the number of pixels of background color to display.
StopBit	The 4-bit StopBit in bits 3:0 defines the position in the memory word that corresponds to the last bit of the last pixel in the tile. The StopBit gives bit resolution to the tile width. When DRAMs are used, the StopBit must be programmed to be consistent with the Bpp defined in the Tile Descriptor. An illegal value causes incorrect display. With VRAMs, the StopBit must be programmed to zero for the first Tile Descriptor. If the Field bit (F) is set to one, these bits are used with the Bpp and StartBit parameters to denote the number of pixels of background color to display.
Fetch Count	The Fetch Count in bits 11:0 specifies the number of bytes minus 2 (Fetch count = Bytes − 2) from the bitmap to be fetched for one horizontal line of the current window tile when DRAMs are used. The fetch count must be an even quantity. With VRAMs, the fetch count must be set to zero for the first Tile Descriptor.
TBLR	The Border Control Bits in bits 15:12, if set to one turn on the border on Top (T), Bottom (B), Left (L), or Right (R) of the window tile when DRAMs are used. All four TBLR bits must be programmed to zero for the first Tile Descriptor when VRAMs are used.
WSt	The 2-bit Window Status (2 bits) in bits 11 and 10 define code presented on the Window Status pins while the window is being displayed. WSt can be used with DRAMs and VRAMs.
PC	The two PC Mode bits in bits 2 and 3 indicate the display format of the current window. The IBM PC CGA Mode (01) allows a bitmap created in IBM PC swapped byte format, which differs from the 82786 format (00). The IBM PC bitmap format positions the least significant byte of a word to the left of the most significant byte on the screen; whereas the 82786 format positions the least significant byte to the right of the most significant byte. For details, refer to Section 3.1.2 "IBM PC Bitmap Format Support." The list below contains valid codes for the PC bits. 00 = 82786 Mode (the default value) 01 = Swapped Byte Mode (IBM PC CGA format) 10 = Swapped Byte, 2 banks (IBM PC CGA format) 11 = Swapped Byte, 4 banks (IBM PCJr format)
Z	The Zoom bit in bit 1, if set, zooms the window using the zoom parameters programmed into the ZoomX and ZoomY Registers (see Sections 3.2.4 "Zoom Support" and 3.3.2 "Display Control Block Registers"). When VRAMs are used and this bit is set, only vertical zoom is obtained, unless external logic is provided.

Table 3-1. Tile Descriptor Parameters (Cont'd.)

F	The Field bit in bit 0 denotes the window tile is the background color and uses the Bpp, StartBit, and StopBit parameters to program the number of pixels of background color to be displayed. If set to one, the Field bit uses the Bpp, StartBit, and StopBit fields (shown in Figure 3-6) as one 12-bit field to denote the number of pixels of background color to be displayed. If zero, the Field bit uses the Bpp, StartBit, and StopBit parameters to define the width of the field tile. The value programmed must be one less than the desired tile width. The Field bit must be set to zero for the first tile when VRAMs are used (see Section 3.2.5).

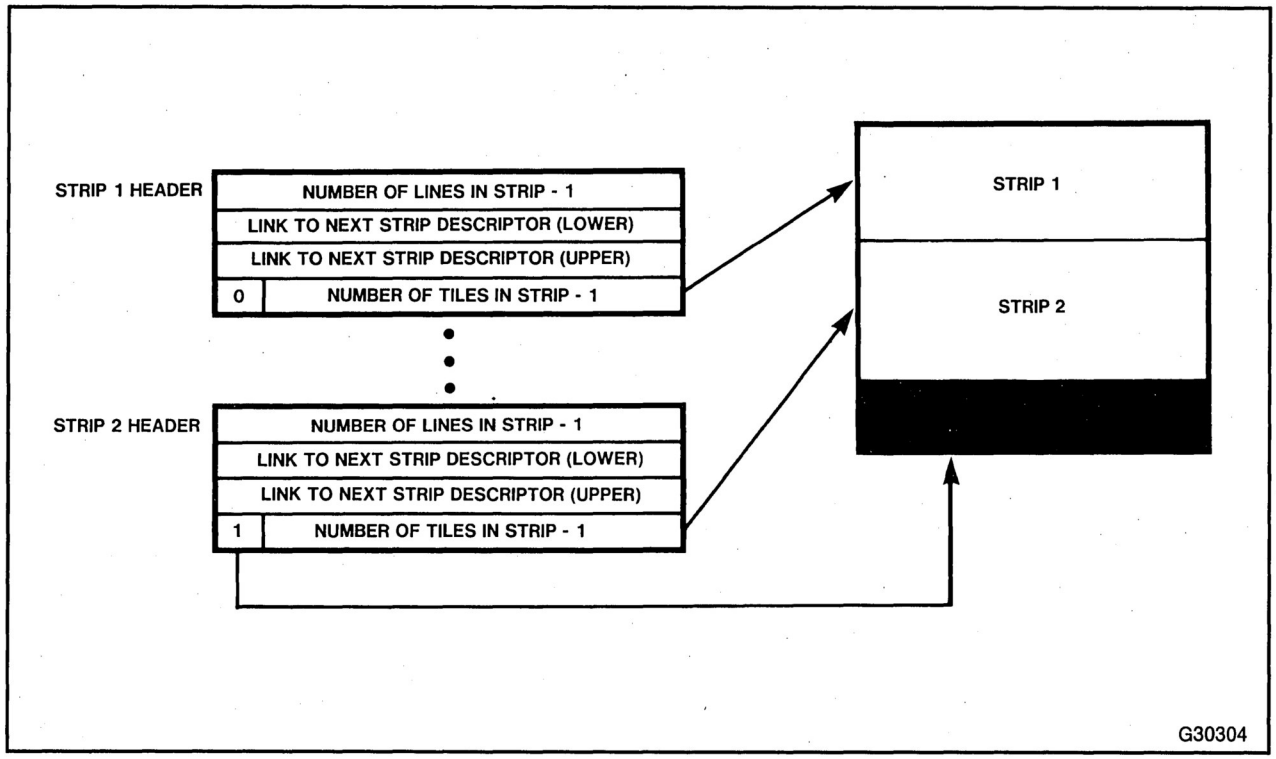


Figure 3-7. C Bit Can Indicate Final Strip Color

3.1.4 Cursor

The Display Processor (DP) supports a block or crosshair hardware cursor. The block cursor can be an 8×8 or 16×16 pixel block. Either cursor can be positioned anywhere on the screen within the defined pixel resolution using the Cursor Mode (CsrMode) Display Control Block Registers to control the cursor. The DP CsrMode Registers also define the type of cursor, block or crosshair; the cursor pattern; and whether it is transparent or opaque. A transparent cursor may be any size and shape up to 16×16 pixels. The hot-spot for the block cursor is the top-left of the cursor shape. The crosshair cursor is one pixel wide and stretches the width and height of the screen. The hot-spot for the crosshair cursor is the intersection of the crosshairs. The cursor color and pattern also are programmable. With frame interrupts to the CPU, the cursor can be programmed off or blinking. For details on frame interrupts, refer to the FRI bit in Section 3.3.1.1 "DPStatus Register and Exception Handling" and the Frint Register in Table 3-8 of Section 3.3.2 "Display Control Block Registers."

In Accelerated Modes the horizontal cursor positioning is reduced to 2, 4, or 8 pixels based on the mode used. For increased pixel resolution or a cursor larger than 16×16, the hardware cursor can be turned off and the cursor can be emulated by software bitblt operations.

3.1.5 Bus Bandwidth Requirements

The system bus bandwidth required by the Display Processor (DP) varies based on the screen parameters such as bits per pixel (bpp) and Video Clock (VClk) frequency. The 82786 has a 40 Mbyte/second bandwidth during block accesses to memory, if the 10 MHz maximum system clock (Clk) and interleaved Fast Page Mode DRAMs are used. The Display Processor uses these fast block reads for screen refresh, which minimizes use of the system bus. For example, with the DP running at its maximum speed of 25 MHz and with 8 bpp, about 65% of the bus is used for display refresh. At 25 MHz and with 1 bpp, the DP requires one-eighth of the bus bandwidth that it required with 8 bpp. Proportional bandwidth reductions also result with reduced VClk frequencies.

3.2 VIDEO INTERFACE

The video interface connects the 82786 to the video display. The 82786 is optimized to drive CRT monitors, but it can also drive other types of displays such as LCDs, plasma, and intelligent printers. This chapter discusses the video interface as as it relates to the Display Processor. Chapter 6 discusses the video interface in detail. The *Intel 82786 Hardware Configuration Application Note* discusses considerations for other types of displays.

The video interface for a CRT depends on the CRT requirements and the resolution and depth (bits per pixel (bpp)) of the image desired. The 82786 can be programmed to generate all CRT signals for up to 8 bpp (256 colors) displays at video rates up to 25 MHz.

Eight parallel Video Data (VDATA) Output lines provide video output. The VDATA output can be used as eight bits per pixel on the CRT, or it can be shifted externally to boost maximum display resolution. An independent Video Clock (VClk) controls the dot rate, which can be up to 25 MHz. Horizontal signals are programmable from 1 to 4096 cycles of the VClk and vertical Synchronization (VSync) signals can be from 1 to 4096 scan lines. With external hardware, a color lookup table can be used, higher display resolution can be achieved by trading off bpp for dot rate, or VRAMs can be used.

Tables 3-2 and 3-3 outline possible display configurations. The calculations assume 60 Hz refresh rate. High resolution CRTs often run slower, which enables the 82786 to generate significantly higher resolutions than these tables depict. All cases assume a CRT horizontal retrace time of 7 μS, except the 512 × 512 × 8 (10 μS) and the 640 × 400 × 8 (13 μS).

Multiple 82786 systems can generate even higher resolutions with more colors. For example, two 82786s can create a noninterlaced, 1144 × 860, 16-color display.

3.2.1 CRT Controller

The Display Processor (DP) CRT Controller operates as a master or a slave based on whether the timing signals, Horizontal Synchronization (HSync), Vertical Synchronization (VSync), and Blank, are inputs or outputs. Table 3-4 outlines typical HSync, VSync, and Blank settings.

When Blank is configured as output, the active display period is determined by the values of VFldStrt, VFldStp, and HFldStrt, and HFldStp. When Blank is configured as input, the external system determines the active display period. The internal video shift register generates video only during the active display period.

Table 3-2. Possible CRT Displays with Standard DRAMs

Bpp	Colors	Noninterlaced	Interlaced
8	256	512 × 512 640 × 400 640 × 480	900 × 675 900 × 675 900 × 675
4	16	870 × 650	1290 × 968
2	4	1144 × 860	1740 × 1302
1	monochrome	1472 × 1104	2288 × 1716

Table 3-3. Possible CRT Displays with VRAMs*

Bpp	Colors	Noninterlaced
8	256	1024 × 1024
4	16	2048 × 1024
2	4	2048 × 2048
1	monochrome	4096 × 2048

* For 64K × 4; with 256K × 4, higher resolutions are supported.

Table 3-4. HSync, VSync, and Blank Settings

HSync and VSync	Blank	Application
Output	Output	Master/Stand-Alone display generated by 82786
Input	Output or Input	82786 generated display superimposed on externally generated video or Slave 82786s in a multiple 82786 system

The CRTMode Display Control Registers (see Table 3-8 in Section 3.3.2 "Display Control Block Registers") define these modes and related parameters such as Horizontal and Vertical synchronization intervals (HSync and VSync), Accelerated Video Modes, and line and frame length.

The DP CRT Controller timing signals HSync, VSync, and Blank can be programmed at a pixel resolution, with a maximum display size of 4096 × 4096 pixels. In any of the Accelerated Display Modes, High-Speed, Very-High-Speed, and Super High-Speed, the horizontal resolution of the CRT timing signals change as outlined in Table 3-5.

These timing signal horizontal resolution changes determine the pixel resolution at which windows and the cursor can be placed. In High-Speed Accelerated Mode (50 MHz), the cursor is zoomed in x and y directions, to appear as a 32×32 pixel blank cursor with 2 pixel resolution on placement and pattern. Use of a software cursor is recommended for Accelerated Modes when exact cursor placement or a larger cursor is essential. By turning off the cursor with the Cursor_On (C) bit in the Video Status (VStat) Register (see Table 3-8 in Section 3.3.2 "Display Control Block Registers") and generating additional software to control and position the cursor, you can obtain one-pixel horizontal resolution for the cursor instead of the 2-, 4-, or 8-bit pixel resolution generated in these Accelerated Modes. In all Accelerated Modes, the border width remains 1 pixel wide.

3.2.2 Video Rates

An external Video Clock (VClk) Input clocks the Display Processor (DP) at any frequency between 10 KHz and 25 MHz, based on the needs of the application. Printer applications generally use the lower frequency. CRT screen applications use the higher frequency. In addition to varied VClk frequencies, the DP also supports interlaced, noninterlaced, and interlace-sync displays.

The DP Accelerated Display Modes (see Section 3.2.1 "CRT Controller") allow systems to trade off bits per pixel (bpp) for dot rate to gain greater overall resolution at the cost of reduced color capacity due to less bpp. Table 3-6 outlines possible bpp for dot rate tradeoffs, which assume a corresponding increase in the size and/or resolution of the monitors used.

The increased dot rate also reduces the timing signal horizontal resolution as outlined in Table 3-5 and cursor positioning discussed in Section 3.2.1 CRT Controller. In Accelerated Display Modes, software cursor control and positioning is recommended instead of using the Cursor Mode (CsrMode) Registers, described in Table 3-8 of Section 3.3.2 "Display Control Block Registers."

The DP can display windows at any of the supported pixel depths (bpp) simultaneously. The Bpp parameter in the Tile Descriptor indicates the bpp for a particular tile (see Table 3-1 in Section 3.1.3.2 "Strip Descriptor Format"). In Accelerated Modes, the maximum

bpp is reduced (e.g., in high-speed mode, only 1, 2, or 4 bpp are supported). To optimize memory allocation and bus bandwidth requirements, text can be stored at 1 bpp, and a complex graphic image can be stored at 8 bpp. When the DP displays the 1 bpp text bitmap, the DP pads the high order Video Data (VDATA) output pins using data from the 1Bpp Pad Register in the Display Control Register Block (see Table 3-8 in Section 3.3.2 "Display Control Block Registers").

3.2.3 HSync and VSync Multiplex Window Status

With external logic, the Horizontal Synchronization (HSync) and Vertical Synchronization (VSync) CRT timing pins can be configured to be decodeable Window Status Output pins. Values assigned to the Window Status bits (Wst) in the Strip Descriptor (see Table 3-1 in Section 3.1.3.2 "Strip Descriptor Format") program these pins. While the Display Processor (DP) displays a tile, these pins can output code for tasks such as externally multiplexing in video data from another source, or selecting a palette range for a particular window. External logic must be used to enable VSync and HSync as CRT timing signals when Blank is high, and as decodeable Window Status signals when Blank is low. To implement Window Status on these pins, see the W parameter in the CRTMode Register in Table 3-8 of Section 3.3.2 "Display Control Block Registers."

3.2.4 Zoom Support

The Display Processor (DP) allows tiles to be zoomed up to 64 times in the x and y directions. The zoom feature is implemented through pixel replication. The DP ZoomX, ZoomY Register (see Table 3-8 in Section 3.3.2 "Display Control Block Registers") determines how much each window is zoomed. The x and y zoom values are independent.

Table 3-5. Timing Signal Resolution Changes

Display Mode	Resolution (pixels)	Pixels/Second (Dot Rate) (MHz)
Normal	1	25
High-Speed	2	50
Very High-Speed	4	100
Super High-Speed	8	200

Table 3-6. BPP for Dot Rate Tradeoffs

Display Mode	BPP	Dot Rate
Normal	8	25 MHz
High-Speed	4	50 MHz
Very High-Speed	2	100 MHz
Super High-Speed	1	200 MHz

With zoom control centralized in the Display Control Block, all zoomed tiles on a display are zoomed by the same amount. A tile either is zoomed or not zoomed. Zoom offset is not supported; a zoomed pixel must be fully displayed or not displayed at all. This restricts tile placement; a tile cannot be placed such that a zoomed pixel is partially obscured.

Each time the Display Processor displays a new strip, the vertical zoom counter is zeroed, which can result in truncated pixels if a tile is not placed at a zoom factor boundary as Figure 3-8 illustrates.

The zoom feature also can support an external character generator. The external system interprets the VDATA7:0 output pins as a character code instead of an 8-bit color value. The zoom feature enables the character code to be repeated for x pixels horizontally and y pixels vertically. Zoom factors of up to 64 vertically and horizontally enable support of character fonts such as Kanji.

Only even zoom factors are supported in the Y direction in interlaced mode. In addition, when zooming interlaced displays, both descriptor lists (interlaced systems use two descriptor lists, one for each frame) must point to the same place in memory, i.e., they must be identical lists.

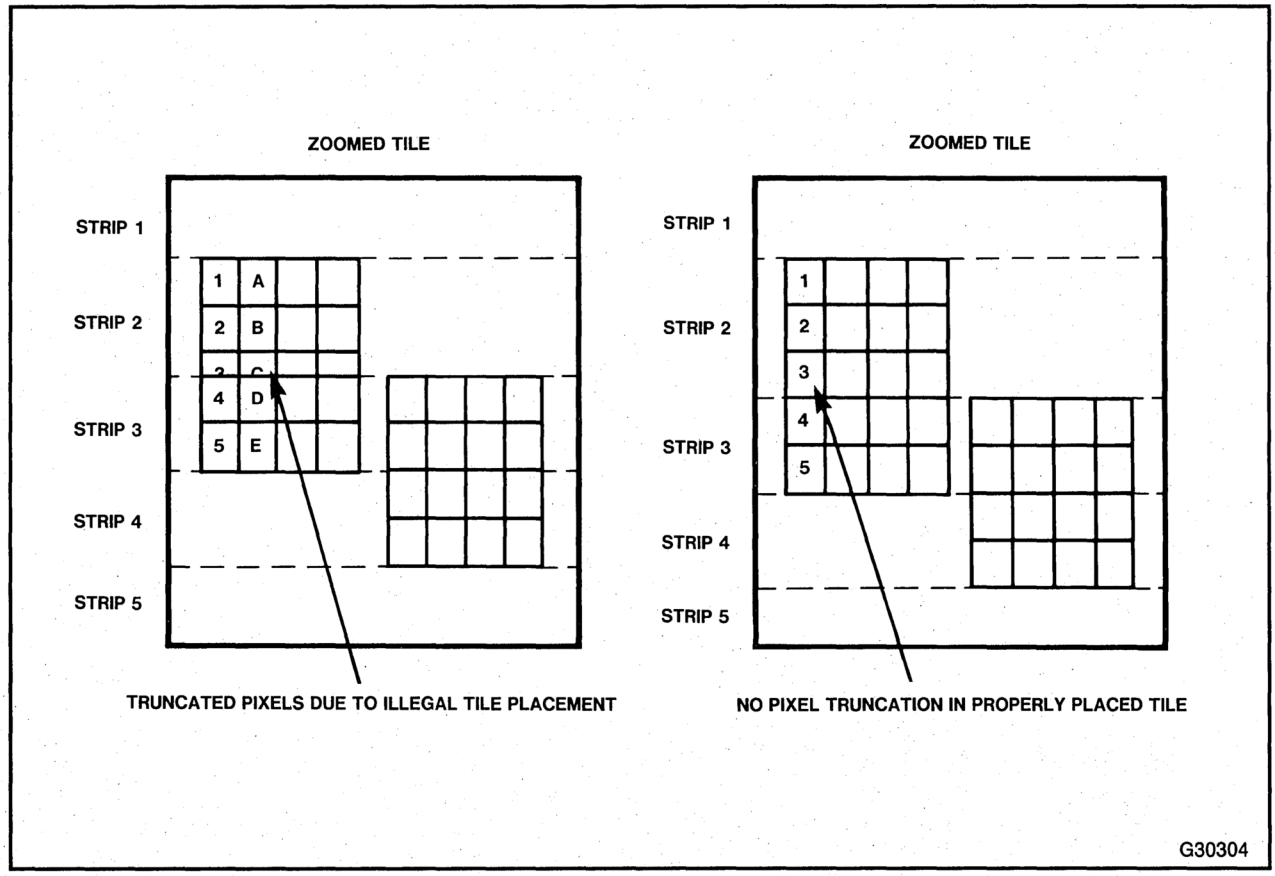


Figure 3-8. Zoom Tile Placement

3.2.5 VRAM Support

The 82786 supports use of dual port video DRAMs (VRAMs) to generate high resolution displays with low system overhead. In VRAM Mode, the 82786 generates

- VRAM control signals to read and write the VRAMs
- Data Transfer (DT) Cycles for loading VRAM data into the VRAM internal shift registers and clocking out data

The video data passes directly from the VRAM shift registers to the video interface and is not brought onto the 82786. As a result, the flexible windowing functions performed by the 82786 Display Processor are not available in VRAM Mode. However, the 82786 supports the concept of hardware overlays, discussed in Section 3.2.5.2.

Table 3-2 summarizes typical screen sizes and resolution attainable using VRAMs. The video data coming from the VRAM Shift Register is serialized externally. Like DRAMs, the screen resolution (size) may be traded off for pixel depth.

When VRAMs are used, a data transfer (DT) cycle executes and loads the first tile for every scan line into the shift register. The second tile and any subsequent tiles for all strips use the VDATA pins. The Window Status pins can be used to multiplex the VRAM video stream and the VDATA stream. The Strip Descriptor determines the address of the DT cycle. The Byte Enable Low ($\overline{BEN}$) pin is used as a DT pin for this case. If the graphics memory banks are interleaved, both banks are loaded in the DT cycle. During Blanking, default video data retrieved from the Default Video Internal Register (see Section 3.3.1) appears on the VDATA pins. The first strip of a VRAM system must be at least two scan lines.

3.2.5.1 SAMPLE VRAM DESIGN

Figure 3-9 depicts a VRAM sample design in which eight 64K × 4 VRAMs generate a 4 bpp display of up to 2K × 1K pixels. The VRAM Address and the Data and Control pins are connected in the same manner as for DRAMs. The difference between VRAM and DRAM system is the video interface.

On a Data Transfer (DT) Cycle to the VRAMs, the VRAM shift registers are loaded from the selected memory row. As Figure 3-9 illustrates, this presents a 32-bit wide word of video data on the shift out pins. This data is then serialized into eight 4 bpp pixels by four 8-to-1 multiplexers. A simple Mod 8 counter, driven by the dot clock frequency, controls the sequencing of the multiplexer through the eight pixels. The Mod 8 counter should be eight times the 82786 VClk frequency. The Shift Clock (SClk) to the VRAMs also is clocked at VClk frequency, and enabled when Blank is low. The DT cycle occurs during Blank time. SClk is activated during display time. This allows each SClk to clock a new 32-bit word to the multiplexer, providing a dot rate of up to 200 MHz using a 25 MHz VClk/SClk.

The video data from the multiplexers can be sent directly to a monitor, or passed through a color lookup table, and/or digital-to-analog convertor (DAC) as described in Chapter 6.

The 82786 supports both static and dynamic Shift Register VRAMs. Dynamic Shift Register VRAMs (like the NEC uPD41264) have a minimum Shift Clock (SClk) frequency

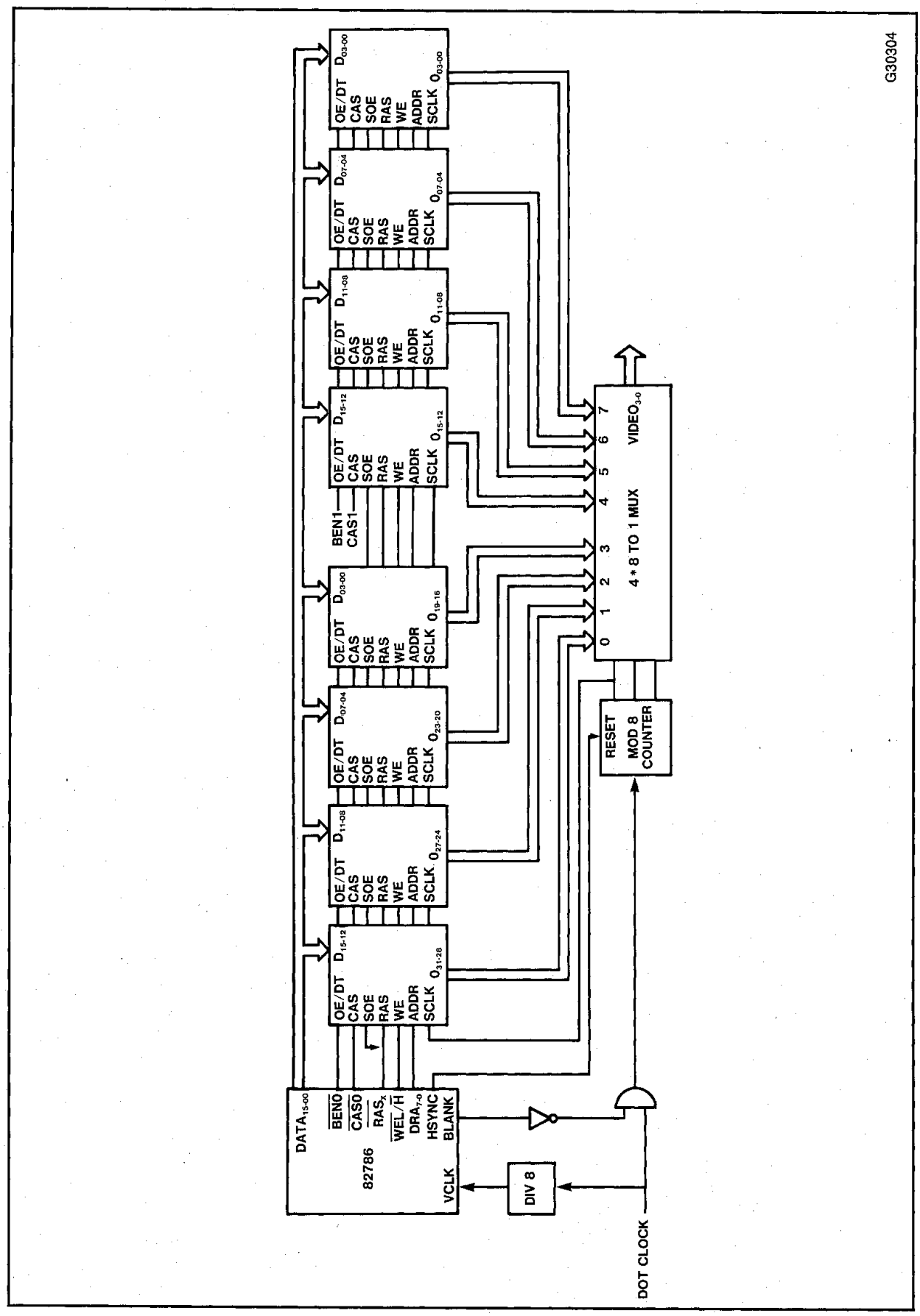


Figure 3-9. Sample VRAM Design

specification below which the data in the Shift Register will decay. Such a specification may be violated during Vertical Blanking. If the 82786 is in Slave Mode, the 82786 never "knows" when Blank will fall and data should be clocked out of the VRAMs. To ensure the VRAM Shift Register stays refreshed, a VRAM Data Transfer (DT) cycle is issued on each Horizontal Synchronization (HSync) during Vertical Blanking. These DT cycles reload the VRAM Shift Register from the same address. After Blank has gone low, the 82786 increments the VRAM Start Address by the Bitmap Width parameter each time the DT Cycle runs (unless zoom is activated).

3.2.5.2 HARDWARE OVERLAYS

Although the 82786 does not currently support hardware windows with VRAMs, the 82786 does support hardware windows overlaid on a VRAM background. This feature provides a useful mechanism to implement pop-up menus or dialog boxes without modifying the contents of the frame buffer and allowing instantaneous addition and deletion of these menus on a very high resolution background.

The hardware overlays are fetched and formatted by the 82786 as hardware windows before being output on the VDATA pins in parallel with the VRAM data. The position and size of the overlays is programmed in the same manner as for DRAMs with Field (F bit) defining the area in which windows are not placed. With additional external logic to select output based on the 82786 Window Status pins, the system can multiplex the VRAM multiplexer output and the VDATA pins. Following vertical blanking, the fetch count for the first display line must not exceed 128 bytes. No restriction exists for subsequent display lines.

3.2.5.3 INITIATING VRAM MODE AND FUNCTIONS

To initiate VRAM Mode, set the VR bit in the BIU Control Register (see Section 4.2.2). In VRAM Mode, the first tile in each display line is a VRAM tile. The address programmed into the VRAM Tile Descriptor (see Section 3.1.3.2 "Strip Descriptor Format") is the address at which the Data Transfer (DT) cycle will be run to the VRAMs, which also indicates the address of the first pixel to be displayed. Only one DT cycle can be run per display line, so the specified address must ensure the pixel data for the display line is contained in one row of memory (256 word for noninterleaved; 512 words for interleaved memory).

Second and all subsequent tiles on a strip can be programmed in DRAM Mode to provide hardware overlays. For hardware overlays on high resolution screens, the DP Accelerated Modes probably will be required to match the DP VDATA pixel rate with the VRAM pixel rate. This restriction is not severe since pop-up menus normally have low pixel depth and the resolution on the placement of the menus is not important. If no hardware overlays are required, a second tile must exist and be programmed to be a field tile defined by the F bit in the Tile Descriptor (see Section 3.1.3.2 "Strip Descriptor Format").

VRAM Mode also supports horizontal split screen operation. As in DRAM Mode, the display is divided into multiple strips. The Strip Descriptor header defines the number of lines in each strip and the link to next strip address. This allows command entry areas to be located in different memory areas from the main frame buffer or various views of the same or different objects to be displayed in different parts of the screen without everything being moved to a common frame buffer.

The 82786 also supports vertical zooming of VRAM windows. Use of the Window Status outputs multiple window pixel depth windows can be displayed on the same screen by controlling the manner in which the VRAM Shift Out data is multiplexed to the video interface. For example, a screen might be divided into three areas: a schematic displayed at the top of the screen at 2 bpp, a piece of IC layout corresponding to the schematic in the middle of the screen at 4 bpp, and a 1 bpp dialog area displayed at the bottom of the screen. Each area of the screen can be derived from VRAM data, together with hardware overlays implementing pop-up menus.

3.2.6 Extended 82786 System Configurations

Extended 82786 systems combine multiple 82786s to provide a greater number of bits per pixel, higher dot rate, larger display area, or more windows. One application for a multiple 82786 system is a high resolution color system, in which three 82786s are used; each 82786 runs 8 bpp to generate one of three colors: red, blue, and green for effectively 24 bpp.

To combine multiple 82786s, the Vertical Synchronization (VSync), Horizontal Synchronization (HSync), and Blank timing signal pins must be configured to allow either one of the 82786s to be the master and the additional 82786s to be slaves or all the 82786s to be slaves controlled by an external synchronization source. For the master, the VSync, HSync, and Blank are configured as outputs as in a standard stand-alone 82786 system (see Table 3-4 in Section 3.2.1 "CRT Controller"). For the slave 82786s, VSync and HSync are inputs. Although VSync and HSync are inputs, they still output Window Status while Blank is low (see Section 3.2.3 "Window Status Bits"). Blank may be programmed independently as input or output based on the application's requirements.

In a multiple 82786 system, the master HFldStrt and HFldStp Register values (see Table 3-8 in Section 3.3.2 "Display Control Block Registers") must be 2 greater than the values in these registers for the slaves.

Each 82786 Display Processor in a multiple 82786 system runs in locked step to allow the outputs to be combined on a single display.

Slave 82786s synchronize to the master every scan line. All slaves run off the same video clock (VClk), system clock (Clk), and RESET. However, because the display data and Strip Descriptor fetch must be synchronized to both VClk and Clk, some of the 82786s tend to go out of synchronization (within a few clock cycles) during the scan line. The slaves are never more than one frame out of synchronization. The impact of the slave 82786s being out of synchronization varies based on the application.

In slave video mode, at least a 1-line vertical front porch and a 7-line vertical back porch are required.

Each 82786 reads data according to its own Slave Enable (SEN) signal. The SEN signal remains active for a short time. A latch must be set for each 82786 when SEN goes high. Send a READY to the CPU when all latches are set, and clear the latch.

When data is read from each 82786, the data is valid only while SEN is high. Latch the data out when SEN is on the falling edge to read the correct data into the CPU. If all

drawing command blocks are identical, except for color and texture, and all the 82786s are started at the same time, all drawing always will be within a few VClk cycles.

For more details on designing and using systems with multiple 82786s, refer to Section 6.5 "Greater Resolution With Multiple 82786s" and the *Intel 82786 Hardware Configuration Application Note.*

3.3 DISPLAY PROCESSOR REGISTERS

The Display Processor (DP) has two types of registers: Internal and Display Control. Six Internal Registers (see Figure 3-10), exist, which can be memory or I/O mapped in the external CPU address space, where the host CPU directly accesses them. The CPU directly addresses the Internal Registers to create command lists, indicate DP status, and provide data for the Video Data (VDATA) Output pins during Blanking. The DP Display Control Registers are local on-chip registers, accessed by the DP Load and Dump Register commands in Section 3.6 and located in a contiguous 42-word Display Control Block (see Section 3.3.2).

3.3.1 Display Processor Internal Registers

The six Display Processor (DP) Internal directly addressable Registers are the Display Processor Opcode Register, three command parameter registers, the DPStatus Register, and the Default Video Register shown in Figure 3-10. For an overview of the DP Internal Registers with all the Internal Registers, refer to Figure 1-3.

The DP Opcode and the three parameter registers are used to send a command to the DP. The DPStatus Register contains the Display Processor's status and is described in the next section. The Default Video Register contains data that appears on the Video Data (VDATA) Output pins during blanking intervals. The CPU can use this register to address an external Palette RAM while loading the palette, which eliminates a separate address path and external logic.

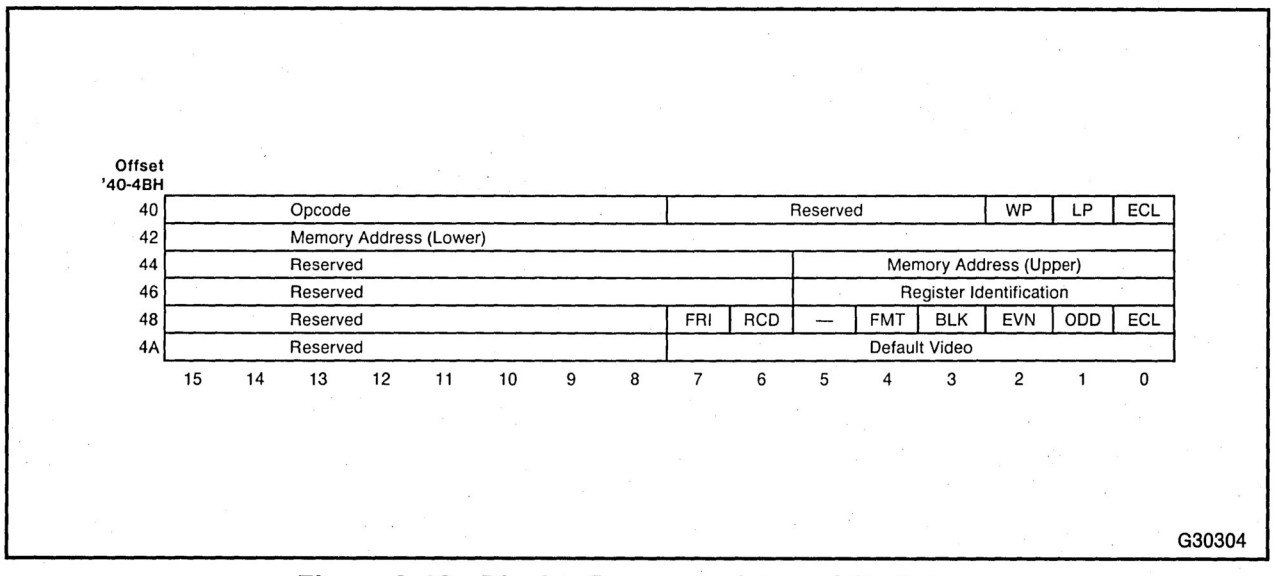
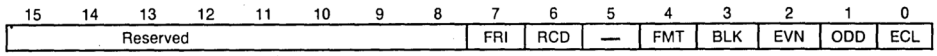


Figure 3-10. Display Processor Internal Registers

3.3.1.1 DPSTATUS REGISTER AND EXCEPTION HANDLING

The Display Processor Status Register (DPStatus) is an 8-bit memory or I/O mapped register. It indicates the current status of the DP and allows interrupt generation based on the state of each status register bit. Figure 3-11 displays the format of the DPStatus Register. Table 3-7 defines the DPStatus Register bits.



G30304

Figure 3-11. DPStatus Register Bits

Table 3-7. DPStatus Register

FRI, Frame Interrupt	The Frame Interrupt bit is set every n frames where n is a value between 1 and 256, which is loaded into the Frint Register (see Table 3-8 in Section 3.3.2 "Display Control Block Registers"). Timing applications use the FRI bit for animation or to set blink rates.
RCD, Reserved Command	The Reserved Command bit is set if the Display Processor (DP) does not recognize the opcode that it is to execute. The DP will not execute the command.
FMT, FIFO Empty	The FIFO Empty bit indicates that the internal Display Processor (DP) FIFO stack is empty. The DP FIFO consists of a 32-double word (32-bits per double word) stack that buffers bitmap data for the DP. When the FIFO empties, this bit is set. An End of Line condition occurs and the rest of the display line displays the background color defined by the FldColor Register (see Table 3-8 in Section 3.3.2 "Display Control Block Registers"). At the beginning of Horizontal Blanking (HBlank), the DP uses the current descriptor to start a new Display Data fetch. A FIFO underrun does not necessarily mean that the whole field is lost; just that the current display line may be corrupted.
BLK, Blank	The Blank bit indicates that the Blank pin is currently high, during vertical blanking.
EVN, Even Field	In Interlace and Interlace-Sync Modes, this bit is set during the even field (Field 1).
ODD, Odd Field	In Interlace and Interlace-Sync Modes, this bit is set during the odd field (Field 2). The Even and Odd status bits help synchronize the 82786 with other interlaced display systems.
ECL, End of Command List	The ECL bit is a copy of the ECL bit in the Opcode Register. It allows the Display Processor (DP) to inform the CPU as soon as the DP completes executing a command.

The Display Processor sets the DPStatus Register End of Command List (ECL) bit only after the DP executes a command and sets the ECL bit in the Opcode Register to 1. The only exception is Loop Mode in which the ECL bit is not set until after Loop Mode is disabled and the DP executes the last command. If interrupts are enabled, an interrupt will be generated after the ECL bit in the DPStatus Register is set.

The ECL bits in the DPStatus register and in the DP Opcode register are always set at the same time. They must be cleared independently, however. The ECL bit in the DP opcode register is cleared by writing a zero to it. The ECL bit in the DPStatus register is cleared by reading the register.

Bit 5 in the DPStatus register is reserved. Be sure the corresponding bit in the Interrupt Mask Register is alwasy masked (set to 1).

3.3.1.2 INTERRUPTS

Interrupts can be masked off using the Display Processor Interrupt Mask Register (IntMsk) (see Table 3-8 in Section 3.3.2 "Display Control Block Registers"), which maps bit-for-bit onto the DPStatus Register. All active interrupts are OR'ed together to drive a single 82786 interrupt line. After being set, the interrupt line remains active until the Status Word in the DPStatus Register and the BIU Control Register are read. The active bits in the DPStatus Register (bits with 0 in the corresponding bit in the Interrupt Mask) are reset to zeroes after the DPStatus Register is read as is the DI bit in the BIU Control Register (described in Section 4.2.2).

3.3.2 Display Control Block Registers

The Display Control Register Block in Figure 3-12 contains 42 16-bit Display Control Registers. The Display Control Block is divided into three functional sections: Video Status (VStat) Registers, CRT Mode (CRTMode) Registers, and Cursor Mode (CsrMode) Registers. Table 3-8 describes these Display Control Registers and their parameters. All reserved fields must be programmed to zeroes.

3.3.3 Pad Registers

The Pad Registers, 1Bpp, 2Bpp, and 4Bpp, supply data presented on the high order Video Data (VDATA) Output pins when the bitmap is less than 8 bits per pixel (bpp). Figure 3-14 illustrates the format of the VDATA pins when pad registers are used with various bits per pixel (bpp) and Accelerated Modes.

3.4 DISPLAY COMMAND SYNCHRONIZATION

To ensure a clean display, without updates occurring during data display, the Display Processor (DP) only loads Display Control Registers during the Vertical Blanking (VBlank) Interval. This synchronizes parameter updates with display refresh.

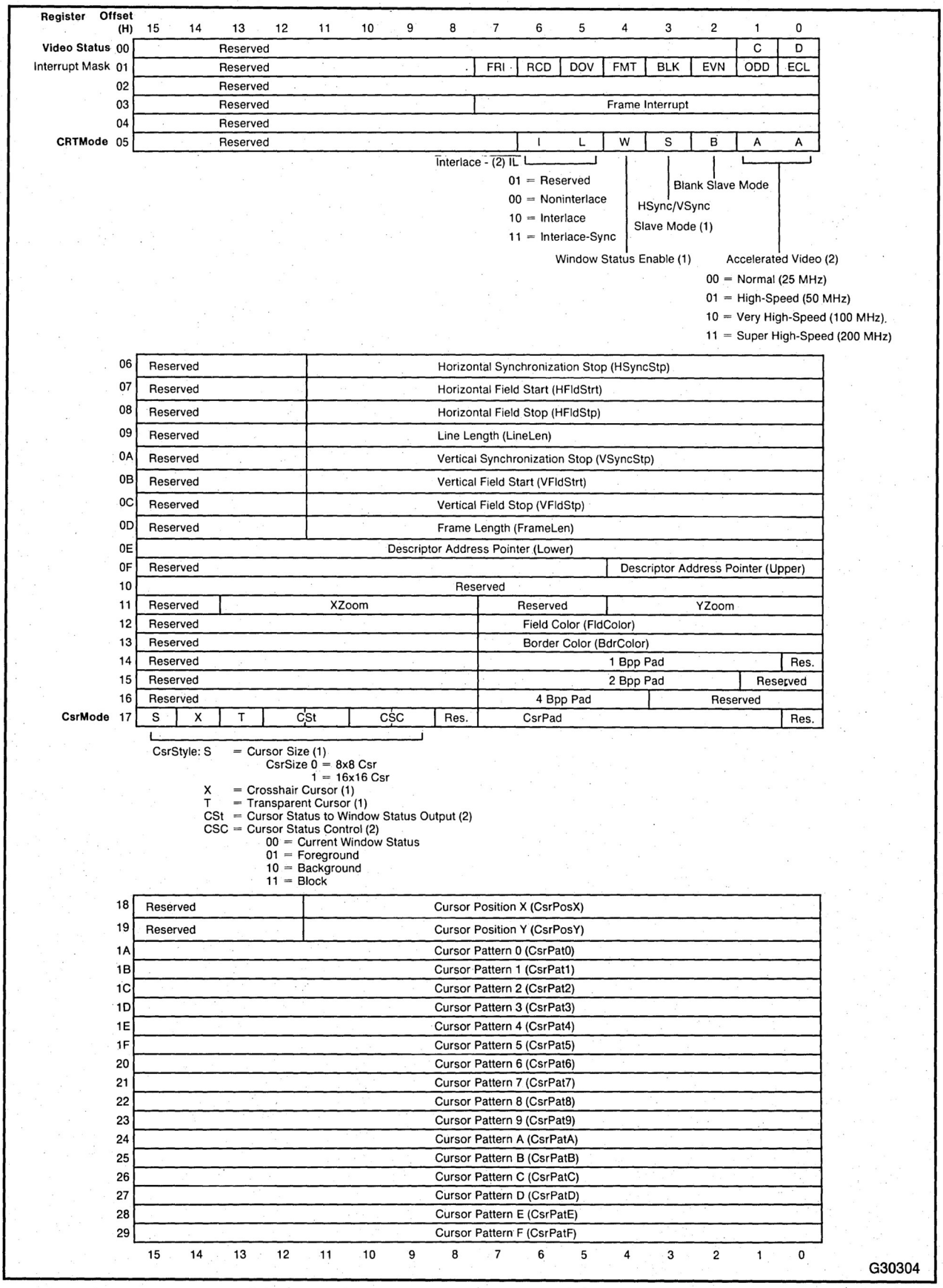


Figure 3-12. Display Control Block Registers

Table 3-8. Display Control Registers

Mnemonic, Register Name
VStat, Video Status VStat has two parameters: CsrOn(1) and DspOn(1). If set when using DRAMs, they turn on the cursor (CsrOn) and/or the display (DspOn).
IntMsk, Interrupt Mask Disables an 82786 interrupt when the corresponding bit in the DPStatus Internal Register is set. A 0 for any bit enables the interrupt. IntMsk is the Display Processor (DP) interrupt Mask and differs from the Interrupt Mask for the Graphics Processor.
Frint, Frame Interrupt The number of frames elapsed between successive setting of the Frint bit in the Internal DPStatus Register. When programming this register, enter the desired value minus 1. The maximum value is 255. The Frame Interrupt counter is reset when a new value is loaded into Frint.
Reserved Reserved for future use. Use of this register will produce unpredictable results. This register should be zeroed to ensure future compatibility.
CRTMode, CRT Controller Modes The CRTMode Register has five parameters: IL(2), W(1), S(1), B(1), and AA(2) parameters for controlling various CRT Controller modes as outlined below. **IL** The Interlace Control Bits define interlace as follows: **00** Display is Noninterlaced. **10** Display is Interlaced (displays the even lines (Field 1) of the frame first, then the odd lines (Field 2)). **11** Display is interlace-sync (similar to interlace, except the odd field display is identical to the even field display). In Interlace Mode, two fields are required to display one frame. One field displays the even lines and the other displays the odd lines. To maintain a line resolution on vertical positioning of windows, a double-length Descriptor Table is required. The first Descriptor Table contains window position information for the first field of the frame; the second Descriptor Table contains information for the second field. This also allows the generation of screens in which different pictures are displayed on the odd and even fields. Command execution occurs at frame boundaries, not field boundaries. The command execution frequency typically is 25/30 Hz instead of the noninterlaced 50/60 Hz. **W** Window Status Enable bit defines use of VSync and HSync timing pins as follows: **0** HSync and VSync operate normally **1** the Window Status Code programmed into the Tile Descriptors will be output on VSync and HSync pins while window data is displayed.

Table 3-8. Display Control Registers (Cont'd.)

Mnemonic, Register Name
VSync represents the most significant bit (MSb) and HSync is the least significant bit (LSb) of the Window Status Code defined in the Tile Descriptor (see Table 3-1 in Section 3.1.3.2 "Strip Descriptor Format"). **S** The HSync/VSync Slave Mode bit. Its value defines whether the Display Processor (DP) operates as a slave or master. If S is 0, the DP operates as a master and the VSync and HSync pins are outputs. If S is 1, the DP is a slave and the VSync and HSync are inputs. In Slave Mode with Window Status enabled, VSync and HSync still become outputs while Blank is low. **B** The Blank Slave Mode bit defines whether Blank is an input or output. If B is 0, the Blank pin is an output and the VFldStrt, VFldStp, HFldStrt, and HFldStp parameters determine the active display period. If B is 1, Blank is an input and the external system determines the active display period. **AA** The Accelerated Video Control Mode bits. With an external latch or shift register, 50, 100, or 200 MHz video data can be generated. In the Accelerated Video Modes, each memory byte represents 2, 4, or 8 physical pixels (see Section 3.2.1 "CRT Controller"). The upper bit(s) of each byte represents the pixels that appear on the left on the display medium. If VRAMs are used, these bits must be set to zero for the first tile. For bit patterns output by these modes, refer to Figure 3-14 "Video Data Pin Outputs." **AA Value** — **Accelerated Mode** 00 — None, Normal (up to 25 MHz) 01 — High-Speed (up to 50 MHz) 10 — Very High-Speed (up to 100 MHz) 11 — Super High-Speed (up to 200 MHz)
HSyncStp, Horizontal Sync Stop The number of Video Clock (VClk) cycles that Horizontal Synchronization (HSync) lasts. Figure 3-13 graphs CRT timing signals. When programming this register, enter the number of cycles minus 3.
HFldStrt, Horizontal Field Start Defines the number of Video Clock (VClk) cycles between the rising edge of Horizontal Synchronization (HSync) and the falling edge of Blank, which is the start of Video Data. When programming this register, enter the number of VClk cycles minus 3.
HFldStp, Horizontal Field Stop Defines the number of Video Clock (VClk) cycles between the rising edge of Horizontal Synchronization (HSync) and the rising edge of the next Blank, which is the end of Video Data. When programming this register, enter the number of VClk cycles minus 3.
LineLen, Line Length The number of Video Clock (VClk) cycles between the rising edge of Horizontal Synchronization (HSync) and the rising edge of the next HSync. When programming this register, enter the number of VClk cycles minus 3.
VsyncStp, Vertical Synchronization Stop The number of Horizontal Synchronizations (HSyncs) between the beginning of Vertical Synchronization (VSync) and the end of VSync. When programming this register, enter the number of HSyncs minus 1. Note that in noninterlaced mode, VSync rises and falls on the rising edge of HSync. In interlaced and interlace-sync mode, VSync has the same timing as in noninterlace mode at the start of each even field (lines 0, 2, 4, etc.), but is delayed by half LineLen at the start of each Odd Field (lines 1, 3, 5, etc.) as shown in Figure 3-13.

Table 3-8. Display Control Registers (Cont'd.)

Mnemonic, Register Name
VFldStrt, Vertical Field Start The number of Horizontal Synchronizations (HSyncs) between the beginning of Vertical Synchronization (VSync) and the end of Vertical Blanking. When programming this register, enter the number of HSyncs minus 1.
VFldStp, Vertical Field Stop The number of Horizontal Synchronizations (HSyncs) between the beginning of Vertical Synchronization (VSync) and the beginning of the next Vertical Blanking (VBlank). When programming this register, enter the number of HSyncs minus 1.
FrameLen, Frame Length The number of Horizontal Synchronizations (HSyncs) minus 1 between the beginning of Vertical Synchronization (VSync) and the beginning of the next VSync.
Descriptor Address Pointer (Lower) The least significant bits of the address of the first Strip Descriptor for the display. After fetching the first Strip Descriptor, the Display Processor (DP) uses the Link Address in the Strip Descriptor Header to fetch the next Strip Descriptor (see Section 3.1.3.2 "Strip Descriptor Format"). The Strip Descriptor Address must be an even byte address.
Descriptor Address Pointer (Upper) The most significant bits of the Descriptor Address Pointer, described above and in Section 3.1.3.2 "Strip Descriptor Format."
Reserved Register Reserved for future use. Use of this register will produce unpredictable results. This register should be zeroed to ensure future compatibility.
ZoomX, ZoomY The X-zoom and Y-zoom factors for the zoomed windows when DRAMs are used. The factor can be any integer number from 1 to 64. When programming this register, enter the factor minus 1. When VRAMs are used, the VRAM data can be zoomed only in the y dimension, unless additional logic is added to handle the x-zoom.
FldColor, Background Color An 8-bit value defining the color of the background displayed when DRAMs are used and in the absence of windows.
BdrColor, Border Color An 8-bit value defining the border color displayed inside selected windows when DRAMs are used.
1Bpp Pad, One Bit Per Pixel Pad Register Creates an 8-bit value by concatenating 1 bit of video data from a 1-bpp bitmap with 7 bits of the Pad Register for the upper 7 bits of video data when DRAMs are used; not applicable to VRAMs.
2Bpp Pad, Two Bit Per Pixel Pad Register Creates an 8-bit value by concatenating 2 bits of video data from a 2-bpp bitmap with 6 bits of the Pad Register for the upper 6 bits of video data when DRAMs are used; not applicable to VRAMs.
4Bpp Pad, Four Bit Per Pixel Pad Register Creates an 8-bit value by concatenating 4 bits of video data from a 4-bpp bitmap with four bits of the Pad Register for the upper 4 bits of video data when DRAMs are used; not applicable to VRAMs.

Table 3-8. Display Control Registers (Cont'd.)

Mnemonic, Register Name
CsrStyle:S(1) X(1) T(1) CSt(2) CSC(2), CsrPad, Cursor Mode The Cursor Mode Register contains two bytes: Cursor Style (CsrStyle), which contains five parameters defining cursor size, type, transparency, status, and control; and Cursor Pad, which concatenated with the VStat Cursor (C) defines cursor color. Cursor Style (CsrStyle) parameters are outlined below. The cursor is displayed directly only for DRAM-derived screens. For VRAM screens, the Cursor Status bits can be used to multiplex the on-chip cursor into the video stream using external logic. In VRAM Mode, the cursor can be derived from either the 82786 or external logic. **S** is the Size bit. If S is 0, the Display Processor (DP) displays an 8×8 pixel cursor. If S is 1, the DP displays a 16×16 pixel cursor. **X** is the Crosshair Mode bit. If X is 0, the Display Processor (DP) displays a block cursor. The Cursor Pattern Registers (CsrPat0:F) contain the cursor pattern. The block cursor hot-spot is the top-left. If X is 1, the DP displays a 1-pixel wide crosshair cursor, which stretches the full height and width of the display. Its hot-spot is the intersection of the crosshairs. Horizontal and vertical cursor pattern resolution and horizontal placement resolution decreases in Accelerated Modes. At 50 MHz High-Speed Mode, the decrease is from one pixel to two pixels. A 16×16 block cursor becomes 32×32. A crosshair cursor becomes 2 pixels wide vertically and horizontally. At 100 MHz Very High-Speed Mode, the horizontal placement and pattern resolution is reduced to 4 and 8 pixels, respectively. Refer to Section 6.4.3 "Cursor Control" for more detail. **T** is the Transparent Mode bit. If T is 0, the cursor is opaque. The concatenation of the 7 most significant bits (MSbs) of the CsrPad Register with 1 determines the cursor foreground color. The concatenation of the 7 MSb CsrPad bits with 0 determines the background color. If T is 1, the cursor background reverts to the bitmap data that is being displayed "behind" the cursor. The T bit does not affect crosshair cursors. See the CsrPad Register below. **CSt** is the Cursor Status. It consists of two programmable bits that are output on the Window Status Output pins while the Cursor is displayed. **CSC** is the Cursor Status Control consisting of two programmable bits defining whether the CSt outputs the current window status or when the programmed CSt value is output. CSC indicates CSt can be output whenever the cursor displays background, foreground, or transparent pixels (useful for inverse video) as defined below: **00** Output current Window Status **01** Output Cst during Cursor Foreground Display **10** Output Cst during Cursor Background Display **11** Output Cst during Cursor Block Display **CsrPad** is the Cursor Pad Register. Although an 8-bit value, only the high 7 bits define the cursor color. The high 7 bits are concatenated with the Cursor Pattern (CsrPat0:F) bits based on the value of the T bit to define the cursor color for each pixel. The sample array below shows how to derive concatenated values and the role of the T bit. **Cursor Pattern Bit Value** **1** / **0** **T = 1**: CsrPad + 1 / Use bitmap data **T = 0**: CsrPad + 1 / CsrPad + 0
CsrPos X, Cursor X-Position Register The Cursor X-Position Register defines the position of the cursor hot-spot relative to the beginning of the line, which is the rising edge of the previous Horizontal Synchronization (HSync). When programming this register, enter the value minus 2.

Table 3-8. Display Control Registers (Cont'd.)

Mnemonic, Register Name
CsrPos Y, Cursor Y-Position Register The Cursor Y-Position Register defines the position of the cursor hot-spot relative to the beginning of the frame, which is the beginning previous Vertical Synchronization (VSync). When programming this register, enter the value minus 1.
CsrPat0:F, Cursor Pattern Registers CsrPat0:F are the 16 Cursor Pattern Registers containing the cursor pattern to be displayed. CsrPat0 is the top row of the cursor. The most significant bit (MSb) is the left bit of the cursor. For an 8×8 cursor, the high byte of the first 8 Cursor Pattern Registers are the cursor pattern.

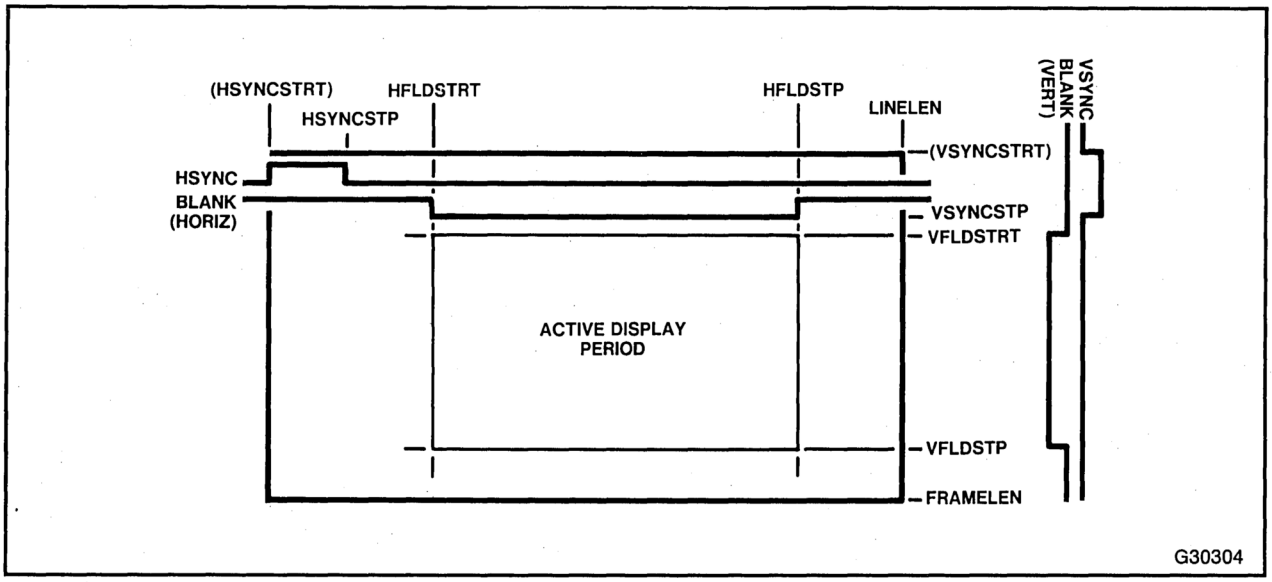


Figure 3-13. CRT Timing Signals

With the Frame Interrupt (FRI) bit in the DPStatus Register and the Frame Interrupt (FRINT) Register, the DP can be programmed to generate a Frame Interrupt once per specified number of frames, which is useful for implementing blinking, scrolling, panning, or other periodic functions. For details on the Frame Interrupt Bit in the DPStatus Register, see Section 3.3.1.1 "DPStatus Register and Exception Handling" and for the Frame Interrupt Register, see Table 3-8 in Section 3.3.2 "Display Control Block Registers."

3.5 COMMAND EXECUTION

At the beginning of each Vertical Sync (VSync) Interval, the Display Processor (DP) checks the End of Command List (ECL) bit in the DP Opcode Register (see Section 3.3.1 "Display Processor Internal Registers"). If the ECL bit is 1, the DP status remains unchanged. If the ECL bit is 0, the DP executes the command. Only one command can be executed per frame.

When the DP completes a command, the DP sets the ECL bit to 1 to tell the CPU that a new command can be written into the Opcode Register. This handshake prevents the CPU from writing a new command before the old one finishes executing.

Normal Display Mode (25 MHz max)

Bpp	Video Data Pins 7	6	5	4	3	2	1	0
1	P17	P16	P15	P14	P13	P12	P11	B00
2	P27	P26	P25	P24	P23	P22	B01	B00
4	P47	P46	P45	P44	B03	B02	B01	B00
8	B07	B06	B05	B04	B03	B02	B01	B00

High-Speed Display Mode (50 MHz)

Bpp	Video Data Pins 7	6	5	4	3	2	1	0
1	P17	P16	P15	B00	P13	P12	P11	B01
2	P27	P26	B01	B00	P23	P22	B11	B10
4	B03	B02	B01	B00	B13	B12	B11	B10

Very High-Speed Display Mode (100 MHz)

Bpp	Video Data Pins 7	6	5	4	3	2	1	0
1	P17	B00	P15	B10	P13	B20	P11	B30
2	B01	B00	B11	B10	B21	B20	B31	B30

Super High-Speed Display Mode (200 MHz)

Bpp	Video Data Pins 7	6	5	4	3	2	1	0
1	B00	B10	B20	B30	B40	B50	B60	B70

Key

Pxy = bit y of the Pad Register for x bpp case

Bmn = Bit n of pixel m in memory. B0 is the first, left-most pixel on the screen fetched from the bitmap.

G30304

Figure 3-14. Video Data Pin Outputs

3.6 DISPLAY PROCESSOR REGISTER COMMANDS

The Display Processor (DP) has the following four register commands, which let you read from or write to one or all of the DP Display Control Block Registers.

- Load Register
- Load All Registers
- Dump Register
- Dump All Registers

Each command and its format is described in the following subsections.

3.6.1 Load Register (LD RG)

Opcode = 0400 hex

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	0	0	Reserved					WP	LP	ECL
Load Address (Lower)															
Reserved													Load Address (Upper)		
Reserved													Register Identification		

The Load Register command loads a pair of registers with the values stored in memory locations starting at the Memory Address. The offsets in Figure 3-12 (Section 3.3.2 "Display Control Block Registers") indicate the Register ID for each register pair. Use this command to update individual pairs of registers, such as the Cursor Position Registers. The address must be an even byte address and the register ID must be an even number. Reserved bits should be programmed to zero to ensure future compatibility.

3.6.2 Load All (LD ALL)

Opcode = 0500 hex

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	0	1	Reserved					WP	LP	ECL
Load Address (Lower)															
Reserved													Load Address (Upper)		

The Load All command loads the entire block of registers in a block read, which starts at the specified Memory Address. The address must be an even byte address. Reserved bits should be programmed to zero to ensure future compatibility.

3.6.3 Dump Register (DMP RG)

Opcode = 0600 hex

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	1	0	Reserved					WP	LP	ECL
Dump Address (Lower)															
Reserved													Dump Address (Upper)		
Reserved													Register Identification		

The Dump Register command tells the Display Processor to write the contents of the specified register pair (indicated by the Register ID) to the memory location defined by Memory Address. The Register ID must be an even number. Use this command to debug and test your 82786 system. Memory Address must be an even byte address. Reserved bits should be programmed to zero to ensure future compatibility.

3.6.4 Dump All (DMP ALL)

Opcode = 0700 hex

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	1	1	Reserved					WP	LP	ECL
Dump Address (Lower)															
Reserved													Dump Address (Upper)		

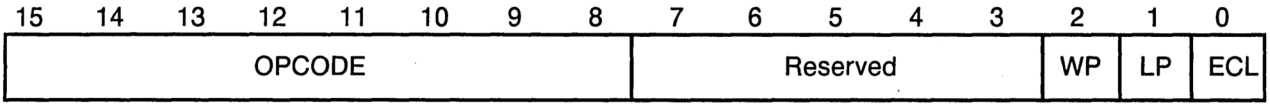
The Dump All command tells the Display Processor to write its entire register block out to a block in memory, starting at the Memory Address. The write consists of a series of single write cycles. Memory Address must be an even byte address. Reserved bits should be programmed to zero.

3.6.5 Loop Command

A Loop Mode is provided on the Display Processor which causes the DP to automatically execute its Opcode every VSYNC. This ensures that the DP parameter registers are kept constantly updated without the need for the CPU to write the Opcode Register each time after first checking that the DECL bit is 1.

3.6.5.1 UPDATING DISPLAY PROCESSOR REGISTERS WITH LOOP MODE

Loop Mode is enabled by writing a 1 to bit 1 of the Opcode Register.



DP Opcode Register
(offset 40h in 128-byte Internal Register Block)

Before writing the Loop Mode bit, the CPU must first poll the ECL bit to ensure that the DP is not currently executing a command. When the ECL bit has been set to 1 by the DP, the CPU then writes the desired Opcode into the upper byte of the Opcode Register, and writes 2 (Hex) into the lower byte (Loop = 1, ECL = 0).

The DP will now execute the Opcode at the start of each VSYNC period, without any further CPU intervention. While in Loop Mode, the DP does not set the ECL bit back to the 1 at the end of each execution. Therefore, when it checks the ECL bit at the beginning of each VSYNC, it is always 0, and command execution begins. The user should consider the DP in a continuous execution mode and should not attempt to change the Opcode while the DP is looping.

To exit Loop Mode, the CPU writes the Loop Mode bit to 0 (without altering the values of the other bits in the Opcode Register). The DP will complete any pending command, and will then set the ECL bit back to 1, indicating to the CPU that the DP execution unit is now idle. If the DP is executing a command at the time the Loop Bit is set to 0, it will complete that command, set ECL, and stop execution. If the DP was not executing, it will execute one more command before setting ECL and stopping execution. Note that the CPU should not modify the contents of the Opcode register until the DP has set the ECL bit to 1.

Loop Mode works for all DP commands, so, for instance, if only the cursor position needs to be kept updated, a LOAD_REG command can be written into the Opcode Register. This minimizes the bus bandwidth required to a single two-word block read. (A LOAD_ALL requires two block reads, one of 32 words and one of 10 words; a DUMP_REG requires two single-word writes; and a DUMP_ALL requires 42 single-word writes.) Compared with a frame time, the bus bandwidth required to execute even an LOAD_ALL is insignificant (about 3.3us out of 16.6ms).

The DP Status Register ECL bit is a reflection of the Opcode Register ECL bit and is set at the same time the Opcode ECL bit is set. This holds true in Loop Mode, and the Status Register ECL bit will only be set when the DP completes its last execution on exiting Loop Mode. The ECL Interrupt will be generated at this point if enabled.

The "Modeset" effect of the first execution, however, is maintained. If the first command out of Reset is a Loop on LOAD_ALL, the first LOAD_ALL that is executed will enable

the CRT Controller and Video Logic. The DP will then continue to automatically execute LOAD-ALL's on every VSYNC until the CPU resets the Loop Enable bit.

If an illegal, or reserved command is given in Loop Mode, the DP will not execute the command and will set the Reserved Command bit in the DP Status Register, generating an Interrupt if enabled. The DP will then exit Loop Mode and the ECL bit will then be set back to 1. The DP will not attempt to re-execute the command until the CPU takes action to correct the Opcode.

3.6.5.2 SUGGESTED MODE OF OPERATION

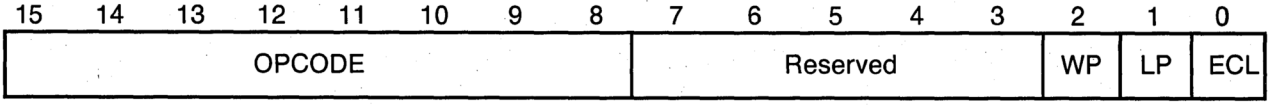
One possible mode of operation is for the CPU to keep a parameter buffer in the form of an array in memory that is an image of the DP's 42-word parameter register block. Updates from a mouse driver or window manager are simply entered into this memory image whenever convenient—for example whenever the CPU polls the mouse driver software. In Loop Mode, this image is then automatically loaded into the DP each VSYNC. This decouples the CPU entirely from the frame timing of the display.

One problem that will arise using a single parameter buffer for updating more than one register (for example, a cursor x and y position) is that in Loop Mode the parameter buffer is always "open" to the DP. It is quite possible that the DP will occasionally read in the parameter buffer between the CPU updating the x and y cursor positions. For one frame then, the display will show a cursor with the new x position, but the old y position.

The way to avoid this, and guarantee clean updates, is to double-buffer the parameter buffer. In this scheme, the CPU maintains two parameter buffers. The CPU updates one buffer while the DP is pointed at the other, and then changes the DP memory-mapped Load Address Register to point the DP at the new buffer. The only restriction is that only one address register must be updated to switch between the buffers, or else the DP may attempt to execute from an incorrect address if only half of the address register has been written. This means that either the two buffers must reside in the same 64K-byte segment of memory (Load Address Register (Lower) is the pointer that switches between the two buffers), or that they must be separated by exactly 64K-bytes (Load Address Register (Upper) is the pointer that switches). Internal logic in the DP ensures that the writing of the Load Address Register cannot collide with the issuance of the Load Address for instruction execution.

3.6.6 Write Protect Bit

Write-Protect for the CRT Parameter Registers is enabled by writing a 1 to bit 2 of the DP Opcode Register.



DP Opcode Register
(offset 40h in 128-byte Internal Register Block)

The Write Protect Bit (bit 2 of the DP Opcode Register) allows the user to write-protect the CRT Timing Parameter Registers. Certain monitors are sensitive to incorrect timing of sync signals that may arise if the user accidentally corrupts the registers. By setting the Write-Protect Bit in the Opcode Register to 1, the CRT Parameter registers are not updated in the LOAD_ALL or LOAD_REG commands.

Write-Protect is not enabled until after the first command given to the DP has been successfully completed. This command must be a LOAD_ALL. Thus, the DP will load the CRT Timing Parameter Registers on the first LOAD_ALL, even if the WP bit is set to 1. Subsequent LOAD commands will not update the CRT Registers. A possible mode of operation is therefore to Reset the 82786 and then give the DP a LOOP on LOAD_ALL with the WP bit set.

The CRT Registers Write-Protected are:

HSyncStop	VSynchStop
HFldStart	VFldStart
HFldStop	VFldStop
LineLength	FrameLength

Note that like the Loop Bit, the Write-Protect Bit must not be changed while the DP is executing or looping. Before changing the Write-Protect Bit, the user should exit Loop if necessary, and wait for the ECL bit to return to 1.

3.7 INITIALIZATION

When the 82786 is RESET, the Display Processor (DP) enters the following state:

- All command execution halts immediately.
- Parameter, Descriptor, or Display Data fetches terminate.
- All Display Outputs VDATA7:0 are reset to default video.
- CRT timing pins HSync, VSync, and Blank are tristated; Display Processor defaults to slave operation. The CRT timing pins stay tristated until the first Load_All command.
- DPStatus word is cleared.
- DP Interrupt Mask is set to all ones to disable all interrupts.
- The DP ECL bit is set to 1.

3.8 VIDEO DATA SIGNATURE ANALYZER

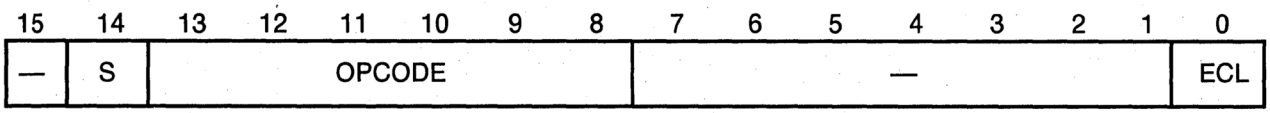
A 16-bit Linear Feedback Shift Register signature analyzer is placed on the Video output bus to compress the video data stream into a single signature that is output onto the Video Data pins during BLANK time. The Signature is also readable by the CPU at the end of a Frame using the DUMP_REG command at Register ID 3C (Read register 3D). This signature analyzer output onto the VDATA pins is activated in DP Test Mode.

3.8.1 Invoking Test Mode

Test Mode is invoked at Reset by driving the RD, WR and MIO pins to the "Test Mode Select" states. These states are sampled by the 82786 when RESET goes inactive. The particular test mode selected is given in the following table:

RD#	WR#	MIO	Mode
0	0	0	RESERVED
0	0	1	RESERVED
0	1	0	DP Test Mode
0	1	1	Force all outputs High
1	0	0	Force all outputs Low
1	0	1	Tristate all outputs
1	1	X	Normal Operation

Once in DP Test Mode, the Signature Analyzer output is enabled by setting bit 14 of the DP Opcode register to 1. (See diagram below.)



Signature Analyzer Output Enable (bit 14)

The remaining bits of the Opcode are still usable as in normal mode, and may be programmed to execute all DP instructions. In Normal Mode, setting bit 14 constitutes an Illegal Opcode, and is trapped as such. The DP will not execute the command, the Reserved Command Status Bit will set, and the DP will continue to operate as before the command had been given.

3.8.2 Operation of the Signature Analyzer

The Signature Analyzer sits on the video data output bus and accumulates the signature during active display time. Outgoing video data is sampled by the signature accumulator which generates a new signature each video clock cycle. At the end of each scan line the signature is output on the video data pins according to the following logic:

IF blank AND hsync# THEN
video_data := 8 LSB of the signature.

IF blank AND hsync THEN
video_data := 8 MSB of the signature.

The signatures from each line are accumulated and the signature of the complete screen is available at the beginning of the VBLANK time. The signature is then cleared on falling VSYNC. In this way there will be a unique signature for each display frame configuration. If the display remains the same for many frames, or if there are certain standard test frames, their signatures will always be the same from one frame to the next, and from one device to

the next. If something changes in the fame—a window is moved or deleted, or a change in the bitmap occurs—the final signature will be different. This allows the signature to be checked at the end of each frame (during VSYNC) for a coarse check on the goodness of the device or system.

The signature is readable by the CPU at the end of a frame using a DUMP_REG command. This is useful as a coarse test to allow the CPU to "watch" the screen when performing a system self-check as part of a diagnostic check. The system may be set up to output a set of test patterns, and the CPU can check the DP signature register at the end of each frame to ensure that the correct signature has been generated.

The signature is not cleared between display lines. If it were, then the final signature would only reflect the goodness of the last display line, and the signatures would have to be checked on every display line. Signatures are accumulated over an entire frame, with the effect that two identical display lines on the same screen will yield two different signatures, since the signature is a result of all the preceding lines.

Video Data is output normally from the Video Data pins in Test Mode exactly as in Normal Mode. The only difference is that during BLANK in Normal Mode, Default Video is output to allow the user to address the Color Palette via the Default Video register, whereas in Test Mode the signature is output during BLANK.

Note that the Signature Analyzer accumulator is always enabled internally regardless of whether the DP is in Test Mode. The Signature Anlayzer Accumulator may therefore be read out by the CPU even in non-test mode and will give useful output. Test Mode simply enables the result of the Signature Anlayzer to be output onto the VDATA pins during BLANK. The value output onto the VDATA pins will be the same value as is read by the CPU.

The Signature Analyzer is implemented using a 16-bit LFSR that generates the primitive polynomial:

$X^{16} + X^{12} + X^2 + X + 1$

A 16-bit LFSR gives a much better fault coverage than a simpler 8-bit LFSR, but even so, there is a significant chance of an error in a typcial display screen, which may contain a quarter-million pixels or more, going undetected. To reduce this error rate, the signature may be checked every display line at the VDATA pins in Test Mode. Note also that the signature only accumulates Video Data that exits on the VDATA pins—in pure VRAM systems the video data does not pass through the 82786, so the signature analyzer will not accumulate this data.

Bus Interface Unit Overview 4

CHAPTER 4
BUS INTERFACE UNIT OVERVIEW

The Bus Interface Unit (BIU) controls all communication between the 82786, the External Bus Master, and memory. The BIU supports three interfaces:

- DRAM/VRAM interface to support accesses to local 82786 graphics memory.
- Slave interface allows CPU access to graphics memory or the 82786 Internal Registers.
- An 80286 external bus for 82786 access to external system memory.

Although, optimized for a host 80286, the 82786 also can be used with the 80386, 80186, or 8086. These CPUs can be tightly coupled with the 82786, directly sharing a number of signals, or they can be more independent and communicate over a system bus such as the MULTIBUS®. Chaining multiple 82786s can provide greater screen resolution or more colors.

The BIU serves as bus translator between the 82786 and graphics and system memory. The BIU controls the majority of the 82786 pins and accepts requests for bus activity from four sources:

- On-chip Graphics Processor (GP)
- On-chip Display Processor (DP)
- Off-chip External Bus Master such as a CPU or DMA
- Graphics DRAM/VRAM Refresh logic in the BIU (see Sections 4.2.3 "DRAM/VRAM Refresh Control Register" and 4.3 "Bus Arbitration."

Possible destinations for BIU bus cycles include:

- Graphics memory directly supported by the 82786. Graphics memory can be accessed by the on-chip Graphics Processor, the on-chip Display Processor, or the off-chip External Bus Master.
- 82786 on-chip memory or I/O mapped Internal Registers, which can be accessed by the External Bus Master only.
- External system memory, which can be accessed by the Graphics Processor or Display Processor only. The External Bus Master also has direct access to external system memory without the 82786.

Block transfers execute at different rates based on the graphics memory configuration and whether the access is to external system memory or graphics memory. Access to graphics memory by the Graphics Processor (GP) or Display Processor (DP) is faster than access by the External Bus Master; the GP and DP access memory directly using the DRAM/VRAM Controller and do not encounter CPU contention. Conversely, the CPU accesses system memory faster than graphics memory; the CPU does not encounter contention from the GP or DP.

The peak data rate varies and can be 10, 20, or 40 MB/Sec. The GP or DP does not know the data rate expected when either initiates a block transfer. The BIU automatically selects the maximum speed that can be achieved for the specific transfer.

4.1 MEMORY STRUCTURE AND INTERNAL REGISTERS

The 82786 address range is 4 MBs, which includes graphics memory directly supported by the DRAM/VRAM Controller and external system memory supported by the host CPU. The 82786 calculates the graphics and external system memory boundary based on the DRAM/VRAM Control Register parameters, which define the graphics memory configuration (see Section 4.2.4). Graphics memory space starts at address 0H and ascends to the amount of graphics memory configured in the BIU DRAM/VRAM Control Register. External system memory occupies the remaining address space, which starts at the top of graphics memory + 1 and can continue to 3FFFFFH. All 4 MBs of address space can be configured to be graphics memory; but if all memory is graphics memory, the 82786 cannot access external system memory.

The External Bus Master uses the 82786 Internal Registers to access graphics memory or instruct the 82786 Graphics Processor or Display Processor to perform specific tasks. All the 82786 Internal Registers are located in a 128-byte contiguous block which starts on an even byte address and is either memory or I/O mapped. Figure 4-1 provides an overview of the 128-byte Internal Register Block. For a detailed view of the Internal Register Block, refer to Figure 1-3.

4.2 INTERNAL REGISTERS

The 128-byte, contiguous, Internal Register Block can be either memory or I/O mapped and is physically distributed between the three 82786 modules: BIU, Graphics Processor (GP), and Display Processor (DP). These directly addressable Internal Registers allow the BIU, GP, and DP to be programmed. The even byte starting address of the Internal Register Block is in the BIU Relocation Register (see Section 4.2.1). Refer to Figure 1-3 for a detailed view of all registers in the Internal Register Block.

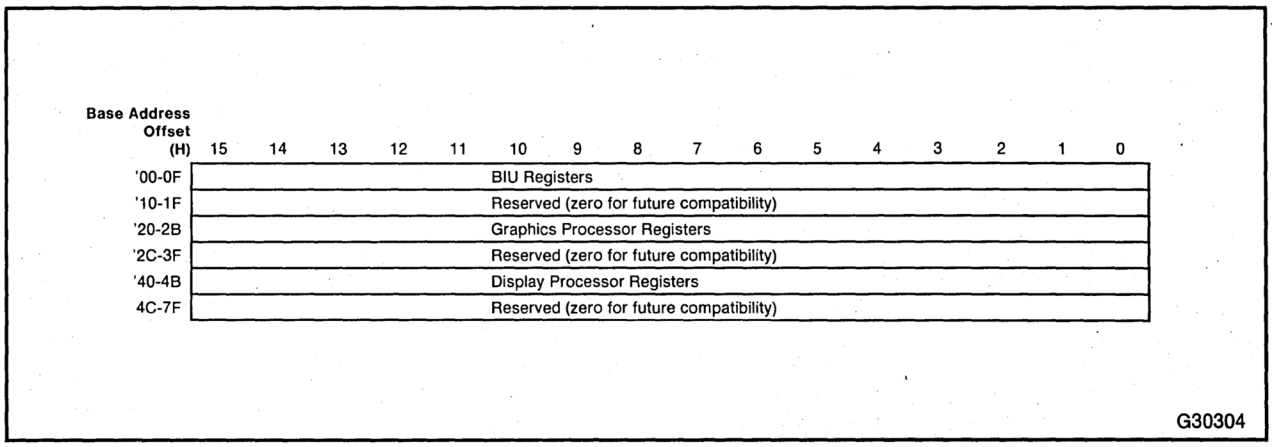


Figure 4-1. 128-Byte Internal Register Block

Figure 4-2 displays a map of BIU Internal Registers. The following subsections discuss details of each BIU Internal Register. For a discussion of the Graphics Processor Internal Registers, refer to Chapter 2, Section 2.2.1 "Internal Graphics Processor (GP) Registers." For details on the Display Processor Internal Registers, refer to Chapter 3, Section 3.3.1 "Display Processor Internal Registers."

All Internal Registers can be either byte addressable or 16-bit word addressable based on the BCP bit value in the BIU Control Register (see Section 4.2.2). If the BCP bit is zero (0), which is its value following RESET, all Internal Registers are byte addressable. Although the 82786 Internal Registers can be accessed a byte at a time, all are considered to be 16 bits, even if all the bits in the registers are not currently used. In 8-bit Mode, the registers must be written in 2-byte even-address pairs. In 16-bit Mode, each register must be written as a 16-bit word. To set 8- or 16-bit address interfaces, refer to Section 4.4.2.2 "8-Bit and 16-Bit Interfaces."

Accesses to Reserved locations have no effect; they execute normally, but can produce indeterminate read data. No register is altered when a write is executed to a Reserved location. To ensure future compatibility, Reserved Registers should always be zeroed.

4.2.1 Relocation Register

The Relocation Register contains the starting location of the Internal Register Block and defines whether the Internal Register Block is memory or I/O mapped. Figure 4-3 defines the format of the BIU Relocation Register.

The Base Address is the upper fifteen bits of the even byte starting address of the 128-byte Internal Register Block. The MIO bit defines whether the Internal Register Block is memory or I/O mapped. If the MIO bit is set to 1, the Internal Register Block is memory mapped. If the MIO bit is set to 0, the Internal Register Block is I/O mapped. For I/O mapping, only 64K of I/O space exists. When setting the base address for I/O mapped registers, the upper six bits of the register (bits 10–15, corresponding to address bits 16–21) must be set

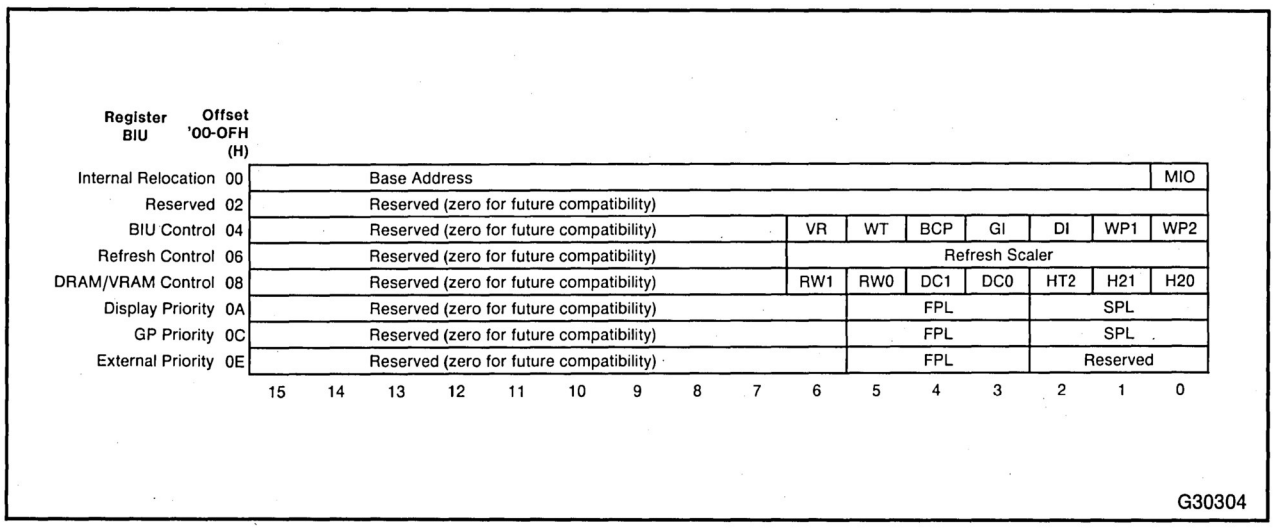


Figure 4-2. BIU Register Map

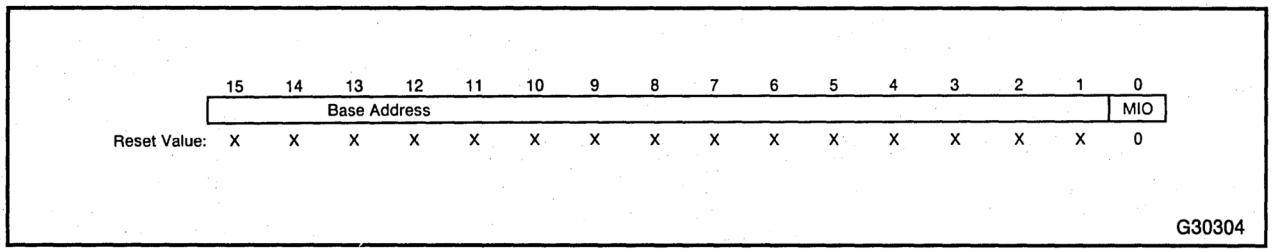


Figure 4-3. Relocation Register

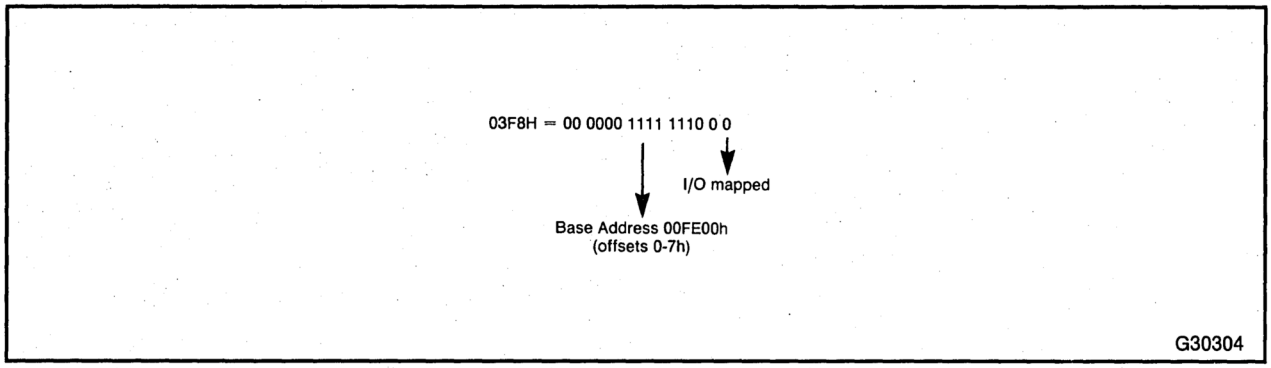


Figure 4-4. Internal Relocation Register Address

to zero (see Figure 4-4). However, when addressing I/O mapped registers, bits 16–21 of the address are don't care. For example, with the value 03F8h written in the Internal Relocation Register (shown in Figure 4-4), the Internal Register Block starts at I/O address FE00h and ends at FE7Fh.

At RESET, the 82786 sets the Base Address for the Internal Register Block to be the entire I/O space whenever Chip Select Low ($\overline{CS}$) is asserted. After RESET, any CPU slave I/O address to the 82786, which activates the $\overline{CS}$ will access the Internal Register Block. During initialization, a write to the Internal Relocation Register should occur to locate the Internal Register Block to a specific even byte memory or I/O address. After the address is written to the Internal Relocation Register, the Internal Register Block occupies 128 bytes of contiguous memory or I/O space starting at the even byte address specified in the Relocation Register, rather than the entire 82786 I/O space.

The Internal Register Block can be located anywhere accessible to the External Bus Master, but if it is memory mapped and configured in graphics memory, the registers take precedence over the memory for CPU accesses to those addresses. Graphics Processor and Display Processor accesses to such addresses will still be directed to DRAM or VRAM.

The even byte starting address written in the Internal Relocation Register determines the memory or I/O address that is placed on the 82786 address pins during access by the External Bus Master to the 82786 Internal Registers. The actual External Bus Master address may differ, and is based on the Chip Select and memory mapping logic.

NOTE

For an 8-bit interface following RESET, write the Base Address into the BIU Relocation Register before any other registers are accessed. For a 16-bit interface, set the BCP bit before writing the Base Address in the Relocation Register.

4.2.2 Control Register

The BIU Control Register defines whether DRAM or VRAM memory cycles fetch display data, byte or word access to the Internal Register Block occurs, Graphics Processor (GP) and Display Processor (DP) Interrupts, and write protection for all BIU Internal Registers. Figure 4-5 outlines the parameters in the BIU Control Register.

VR—The VR bit defines whether the 82786 generates VRAM or DRAM memory cycles to fetch display data. If VR is set to 1, the 82786 generates dual port video RAM (VRAM) memory cycles to generate display data. If VR is reset to 0, the 82786 performs conventional dynamic RAM (DRAM) page mode memory cycles to fetch display data. For information on displaying data, refer to Chapter 3, which discusses Display Processor details.

WT—The WT bit determines the minimum number of wait states possible in a synchronous 80186 interface. If set to 1, a minimum of two or three wait states occur during memory read and write cycles. For details, see Section 4.5.2 "80186 Synchronous Interface."

BCP—The BCP bit determines whether the External Bus Master accesses the Internal Register Block by bytes or words. If the BCP bit is set to 1, 16-bit word accesses occur. If it is set to 0, 8-bit byte accesses occur. Refer to Section 4.4.2.2 "8-Bit and 16-Bit Interfaces" for details.

GI—The Graphics Processor Interrupt bit is set when the Graphics Processor (GP) issues an interrupt. The GI is cleared when the 82786 is RESET or the BIU Control Register is read.

DI—The Display Processor Interrupt bit is set when the Display Processor (DP) issues an interrupt. The Display Processor Interrupt bit is cleared when the 82786 is RESET or the BIU Control Register is read.

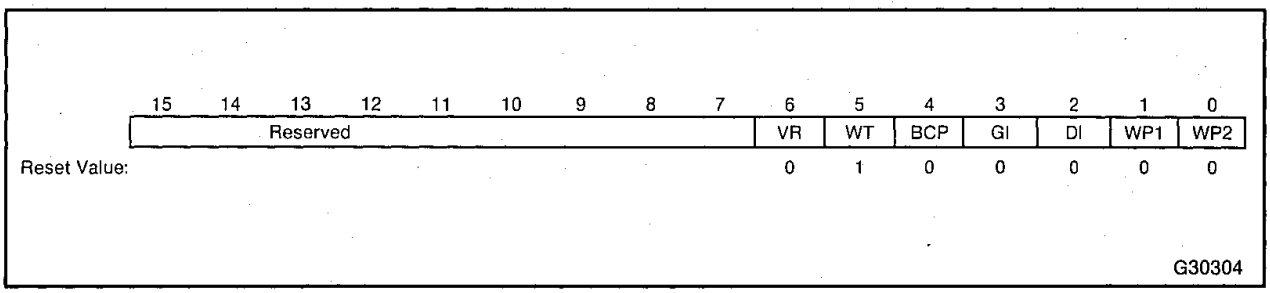


Figure 4-5. BIU Control Register

WP1—The Write Protection One bit, if set to 1, write protects the contents of all BIU Internal Registers except for itself and the WP2 bit in the BIU Control Register.

WP2—The Write Protection Two bit, if set to 1, write protects the contents of all BIU Internal Registers, including the WP1 bit and itself. To regain access to this and all other BIU Registers, the 82786 must be RESET.

4.2.3 DRAM/VRAM Refresh Control Register

The DRAM/VRAM Refresh Control Register contains the Refresh Scalar, which is a 6-bit value defining the frequency of refresh cycles to graphics memory. Figure 4-6 illustrates the format of the Refresh Scaler and its RESET value.

The Refresh Scalar value is internally decremented every 16 clock (CLK) cycles. When the Refresh Scaler generates a borrow, it also generates a graphics memory refresh request. For example, the Refresh Scaler value 18 generates a graphics memory refresh request every 15.2 μS at 20 MHz. Refresh cycles are disabled when the Refresh Scaler is set to all ones (1s). To determine the frequency that DRAMs/VRAMs require refresh, use the following formula:

$$\frac{\text{Tref X CLK}}{\text{16 X Refresh_Rows}} - 1$$

A sample application using this formula assumes a system uses 51C256H DRAMs, which require 256 rows to be refreshed every 4 mS (Tref). These DRAMs consist of 512 address rows of 512 address columns, but for refresh purposes, only 256 row addresses (A0-A7) need to be refreshed within the 4 mS refresh time. The A8 input is not used for refresh cycles. Although the 82786 maintains a full 10-bit refresh address, the upper two bits are not used in this configuration. Also, assume a 10 MHz 82786 CLK. The value for the DRAM/VRAM Refresh Control Register can be calculated as follows:

$$\frac{\text{4 mS X 20 MHz}}{\text{16 X 256}} - 1 = 18.53$$

The result should be rounded down, which gives a value of 18 for the DRAM/VRAM Refresh Scaler. The result is based on the type of DRAM or VRAM used and the 82786 CLK frequency; the configuration of memory chips does not matter.

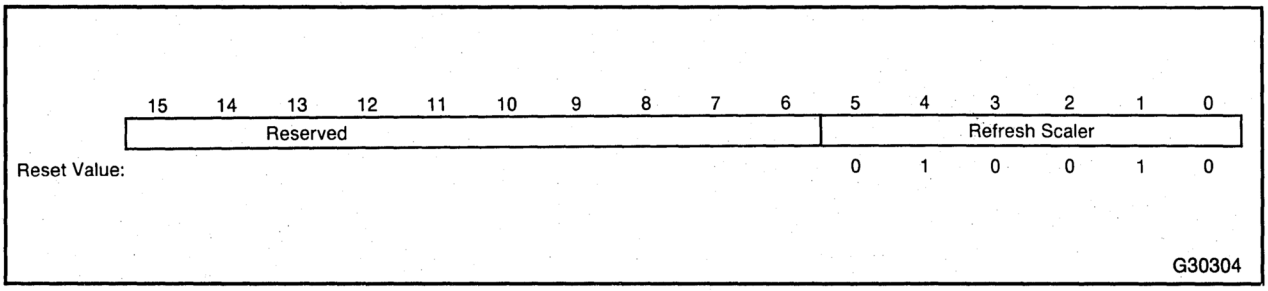


Figure 4-6. Refresh Scaler

A latency time exists between the refresh request and the refresh cycle. The graphics memory refresh always occurs following the completion of the current bus cycle. Graphics memory refresh cycles can interrupt block transfers, but only on doubleword boundaries. The longest latency occurs if a graphics memory refresh request occurs just after the 82786 receives a Hold Acknowledge (HLDA) to begin a Master Mode (see Section 4.4.1.2) block transfer. The 82786 must complete two Master Mode cycles before the graphics memory refresh cycle can occur because a block transfer can be interrupted only on a doubleword boundary. During this latency, additional graphics memory refresh requests can be generated. The 82786 has a refresh request queue, which allows up to three graphics memory refresh requests to be pending. As soon as the bus is free, all queued refresh cycles execute consecutively. With the preceding Refresh Scaler value 18, a graphics memory refresh request is generated every 15.2 microseconds (μS), which can be calculated using the following formula.

$$\text{Request_Time} = \frac{16 \text{ X } (\text{RefreshCount} + 1)}{\text{Clk}}$$

$$\frac{16 \text{ X } (18 + 1)}{20 \text{ MHz}} = 15.2\ \mu\text{S}$$

To determine the amount of latency that the DRAMs/VRAMs will tolerate for each row, use the following formula:

$$\text{Allowed_Latency} = \text{Tref} - (\text{Request_Time X Refresh_Rows})$$

$$= 4\text{mS} - (15.2\ \mu\text{S X } 256) = 108.8\ \mu\text{S}$$

The 82786 capacity to queue three graphics memory refresh requests determines the real latency limit, which can be calculated as follows:

$$\text{Maximum_Latency} = \text{Queue_Size X Refresh_Time}$$

$$3 \text{ X } 15.2 = 45.6\ \mu\text{S}$$

Another key calculation is the maximum number of wait states allowed for an 82786 Master Mode transfer, which can be calculated as follows:

$$\text{Wait State} = \frac{(\text{Maximum_Latency X PClk}) - \text{Overhead})}{\text{bus cycles}}$$

$$\frac{(45.6\ \mu\text{S X } 10 \text{ MHz}) - 7)}{2} = 224$$

In this example, the refresh latency is not a problem. If the system memory caused the 82786 to wait more than 224 wait states for a Master Mode access (see Section 4.5), the DRAM refresh would be missed and the display refresh would be lost.

4.2.4 DRAM/VRAM Control Register

The DRAM/VRAM Control Register contains seven bits defining the memory configuration so the 82786 can locate addresses. The DRAM/VRAM Control Register indicates the number of graphics memory rows, type of configuration, and height of memory components used. Figure 4-7 displays parameters in the DRAM/VRAM Control Register.

RW1:0—The two RW1:0 bits define the number of graphics memory rows as shown in Table 4-1. These bits help define the graphics and external system memory boundary. The RW1:0 bits also disable RAS signals that are not driving DRAMs or VRAMs.

DC1:0—The DC1:0 bits determine the DRAM/VRAM Configuration as shown in Table 4-2. The value of these bits controls the rate of block transfers and orientation of CAS1 and CAS0.

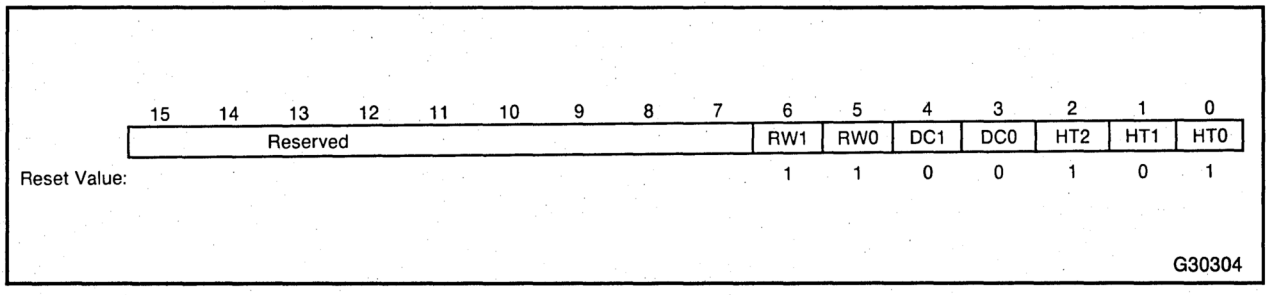


Figure 4-7. DRAM/VRAM Control Register

Table 4-1. RW1:0 Values

RW1	RW0	Definition
0	0	1 Row
0	1	2 Rows
1	0	3 Rows
1	1	4 Rows

Table 4-2. DC1:0 Values

DC1	DC0	Definition
0	0	Page Mode, Noninterleaved
1	0	Fast Page Mode, Noninterleaved
0	1	Page Mode, Interleaved
1	1	Fast Page Mode Interleaved

HT2:0—The HT2:0 bits define the DRAM/VRAM Height (not size) of graphics memory chips in the system as shown in Table 4-3. All DRAMs/VRAMs must be the same height.

NOTE

DRAMs with a height of 64K are not allowed in the graphics memory of a system in which master mode will be used. There is no limitation on the total DRAM density (64K × 4 cannot be used; 256K × 1 is okay).

4.2.5 Display Priority Register

The Display Priority Register determines the priority of Display Processor (DP) requests for the bus and for block transfers. A request is evaluated based on whether it is the first request (FPL) for the bus or a subsequent request (SPL) to complete a block transfer. Figure 4-8 illustrates the format of the Display Priority Register.

The Display Priority Register contains two 3-bit values defining the priority of FPL and SPL requests. Both requests use the same priority scale. The highest priority is 111. The lowest is 000. For a discussion of priorities and their use by the BIU, refer to Section 4.3 "Bus Cycle Arbitration."

Table 4-3. HT2:0 Values

HT2	HT1	HT0	Height
0	0	0	8K × N* Devices
0	0	1	16K × N Devices
0	1	0	32K × N Devices
0	1	1	64K × N Devices
1	0	0	128K × N Devices
1	0	1	256K × N Devices
1	1	0	512K × N Devices
1	1	1	1M × N Devices

* N = DRAM/VRAM width

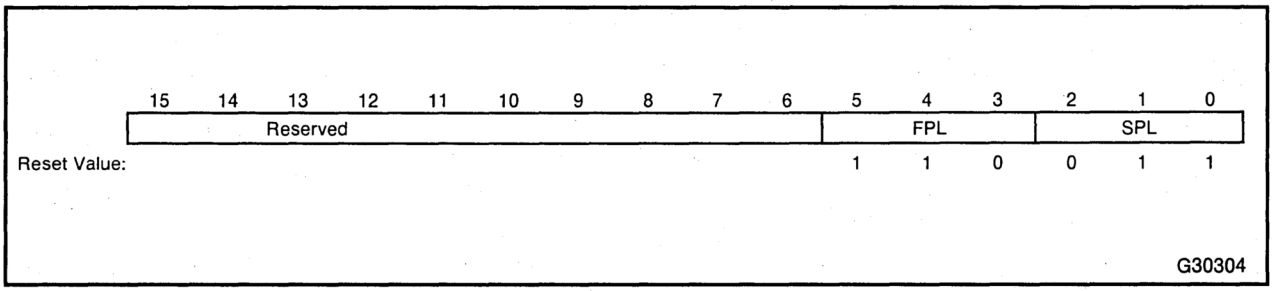


Figure 4-8. Display Priority Register

4.2.6 Graphics Priority Register

The Graphics Priority Register determines the priority of Graphics Processor (GP) requests for the bus and block transfers. A request is evaluated based on whether it is the first (FPL) request for the bus or a subsequent (SPL) request to complete a block transfer. Figure 4-9 illustrates the format of the Graphics Processor Priority Register.

The Graphics Processor Priority Register is divided into two 3-bit values that indicate the priority of FPL and SPL requests. Both requests use the same priority scale. The highest priority is 111. The lowest priority is 000. For a discussion of priorities and their use by the BIU, refer to Section 4.3 "Bus Cycle Arbitration."

The Graphics Processor Priority Register is divided into two 3-bit values that indicate the priority of FPL and SPL requests. Both requests use the same priority scale. The highest priority is 111. The lowest priority is 000. For a discussion of priorities and their use by the BIU, refer to Section 4.3 "Bus Cycle Arbitration."

4.2.7 External Priority Register

The External Priority Register defines the priority of requests from the External Bus Master. The Priority value occupies bits 3 through 5 and 0 through 2 of the low byte as shown in Figure 4-10.

The External Priority Register indicates the priority of External Bus Master requests. The 3-bit priority value occupies bits 3 through 5 in the 16-bit External Priority Register. The highest FPL is 111. The lowest FPL is 000. For a discussion of priorities and their use by the BIU, refer to Section 4.3 "Bus Cycle Arbitration."

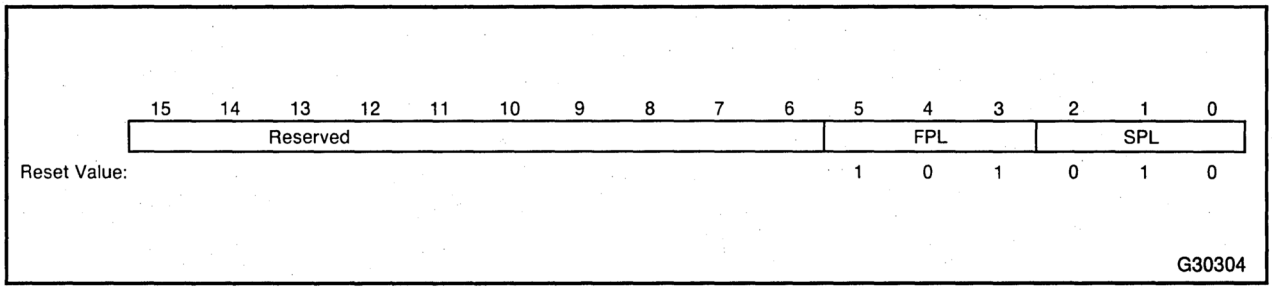


Figure 4-9. Graphics Processor Priority Register

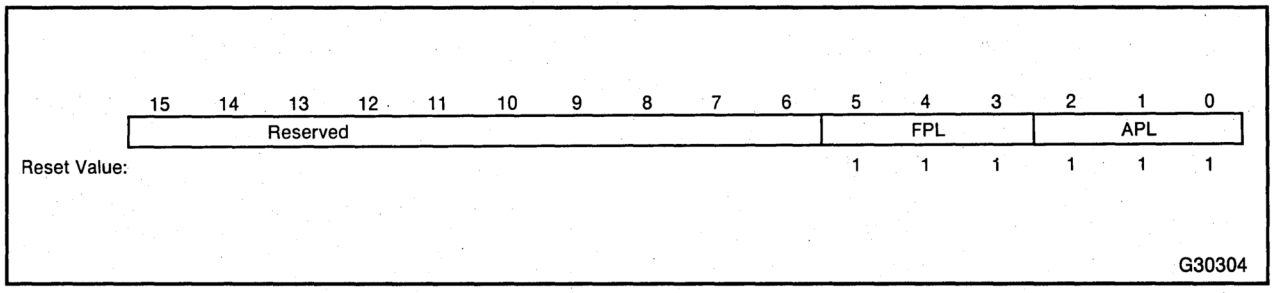


Figure 4-10. External Priority Register

The APL field of the BIU External Priority Register (bits 2–0) is used to define the priority CPU requests assume every 42 pin CLKs. The priority value specified in this field will be selected every 42 pin CLKs and will only switch back to the standard priority value specified in the FPL field in the BIU External Priority Register upon execution of a slave CPU bus cycle.

A CPU may occasionally encounter long delays (> 8us) when accessing the 82786 DRAM or internal registers due to conflicts with display processor bitmap fetching if the DP has highest priority. IBM PC's have a maximum bus latency specified of 2.1us for XT's and 2.5us for AT's. This feature allows the CPU to get a bus cycle every 2.1us. This feature can be "disabled" by programming the new APL field to the same value as the FPL field.

4.3 BUS CYCLE ARBITRATION

The Bus Interface Unit (BIU) receives bus requests from the Display Processor (DP), the Graphics Processor (GP), and the External Bus Master. The BIU arbitrates requests from each of these sources with the same arbitration network based on the programmable priority defined in the Display, Graphics, and External BIU Priority Registers, described in Sections 4.2.5 through 4.2.7. Graphics memory refresh requests, defined by the Refresh Scaler in the DRAM/VRAM Refresh Control Register, have the highest priority request at all times unless the Refresh Scaler is disabled by setting all bits to one.

With priority for a bus request and block transfer request being individually programmable, each block transfer request can have a unique priority. If the block transfer priority is set low, the bus cycle can be interrupted by a higher priority request. A performance penalty occurs when interrupting a block transfer, but the interrupt reduces bus latency. For each system design, the tradeoff between maximum tolerable bus latency and system bus performance must be made. For details on bus latency, refer to the calculations in Section 4.2.3 "DRAM/VRAM Refresh Control Register."

4.3.1 Priority Levels

One of two priority levels is associated with each request from the Display and Graphics Processors: a First Priority Level (FPL) and a Subsequent Priority Level (SPL). The FPL is the priority with which the first request for a bus cycle is arbitrated. The SPL is the priority associated with subsequent requests to complete a block transfer bus cycle. With separate priorities for bus requests and block transfer requests, each block transfer can execute with a different priority level. If a higher priority request occurs while a block transfer is executing, the BIU suspends the current block transfer and acknowledges the higher priority request. After completing the higher priority access, the BIU arbitrates the requests again. The suspended block transfer is arbitrated with its SPL priority for subsequent requests until it completes executing the block transfer. Block transfers can be interrupted on doubleword boundaries only. External requests do not have an SPL or modified priority level because the External Bus Master cannot execute block transfers. Table 4-4 lists the default priorities from highest to lowest following RESET.

Table 4-4. Default Priority Levels Following RESET

Priority	Type	Value
External	FPL	7
Display	FPL	6
Graphics	FPL	5
Display	SPL	3
Graphics	SPL	2

Table 4-5. Suggested Priority Values

Processor	FPL	SPL
Display Processor	6	6
Graphics Processor	2	2
External Processor	4	NA

Each priority is defined by a 3-bit code. The highest priority value is 7 (111); the lowest priority is 0 (000). If two Priority Registers are programmed with the same value, the BIU uses the priority chain below, which lists priorities from the highest to the lowest.

1. Display Processor
2. Graphics Processor
3. External

Normally, request priorities are set during initialization following RESET and write protected using the WP1 or WP2 bit in the BIU Control Register (see Section 4.2.2). Suggested priority values follow in Table 4-5. After the system is working, these values can be adjusted for optimum performance. The optimum values depend on the CPU and video speeds, as well as the CPU and graphics command mix and window arrangement. Usually these values are initialized following RESET and remain the same, but changing these values when an application changes modes may be advantageous for some applications.

4.3.2 Priority Exceptions

The priority scheme outlined in the previous section is overridden when the 82786 receives either a graphics memory refresh request or an External Slave request while the 82786 is executing a bus cycle to the external system memory and is waiting in a Hold Acknowledge (HLDA) loop to acquire the bus from the External Bus Master. A graphics memory refresh request always has the highest priority of all requests and must be executed because the 82786 can be in the HLDA loop for a long time. The External Slave request must be honored to avoid a command lockout situation in which the External Bus Master is waiting for the bus cycle it already started (the External slave request directed at the 82786) to be completed before it responds with a HLDA signal. In some cases, the CPU executes multiple bus cycles to the 82786 before asserting HLDA. The 82786 handles these properly.

4.4 MASTER AND SLAVE INTERFACES

The 82786 can operate as a Master or a Slave. As a slave, the CPU or DMA can perform read and write cycles to the 82786 Internal Registers or graphics memory. As the Bus Master, the 82786 Graphics Processor (GP) and Display Processor (DP) can perform read and write cycles to the host CPU's system memory. The following sections discuss both interfaces and relevant concepts.

4.4.1 Master Mode Interface

The Graphics Processor (GP) and Display Processor (DP) can execute bus cycles as the Bus Master to access memory beyond the limits of configured graphics memory. When the 82786 is the Bus Master, it operates in Master Mode as described in Section 4.4.1.1 "Initiating Master Mode Interface."

The Bus Interface Unit (BIU) distinguishes between graphics memory and external memory by the memory address. Graphics memory address space is defined by the DRAM/VRAM Control Register (see Section 4.2.4). External memory starts at the top of graphics memory space and can ascend to address 3FFFFFH, the maximum 82786 address. Only GP and DP bus cycles can address external system memory.

DRAM's with a HEIGHT of 64K are not allowed in the graphics memory of a system in which Master Mode will be used. There is no limitation on the total DRAM density (64K $\times$ 4 DRAM's cannot be used; 256K $\times$ 1 DRAM's are okay).

4.4.1.1 INITIATING MASTER MODE INTERFACE

The 82786 bus timings are similar to those of the 80286. An 82786 requests the bus by indicating a high level on the Hold Request (HREQ) line to the host CPU or to the bus arbitration logic. The 82786 stays in a Hold Acknowledge (HLDA) loop until the CPU or bus arbitration logic returns a HLDA. The 82786 drives the external bus, Address Lines A21:0, Data Lines D15:0, Read Low ($\overline{RD}$), Write Low ($\overline{WR}$), Master Enable High (MEN), and Byte High Enable Low ($\overline{BHE}$), only after the 82786 receives a Hold Acknowledge (HLDA) from the External Bus Master. The HLDA line can be externally synchronized if $\overline{BHE}$ is high at trailing RESET (82786 Synchronous Interface, see Sections 4.5.1 and 4.5.2) or internally synchronized if $\overline{BHE}$ is low at trailing RESET (asynchronous, see Section 4.5.3). The 82786 indicates that it controls the external bus by a high level on the Master Enable (MEN) output. MEN also controls the direction of the data transceivers between the 82786 and the external bus. The 82786 deactivates the HREQ line and returns control to the External Bus Master when the 82786 no longer requires access to external memory or when it senses an inactive HLDA (see Section 4.4.1.2 "Retaining Control of the System Bus").

Table 4-6 illustrates the performance of an 82786 Master on an 80286 style bus with zero wait states. The HREQ and HLDA overhead is not included in these numbers. The bus cycles are consecutive bus cycles to external memory without relinquishing HREQ. A synchronous, 10 MHz, 82786/80286 pair is assumed. The graphics memory cycle times are included for comparison. The numbers in parentheses are the respective number of bus cycles.

Table 4-6. 80286 Bus Master Performance

Operation	External	Graphics Memory
Single Read/Write	5.0 Mb/s (4)	6.7 Mb/s (3)
Read-Modify-Write	3.3 Mb/s (6)	5.0 Mb/s (4)
Block Transfer	10.0 Mb/s (2)	10, 20 or 40 Mb/s (2, 1, or .5)

The 82786 begins the first Master Mode bus access on the cycle after the HLDA is activated. The only delay is the time between the 82786 activating HREQ and the system releasing the bus and returning the HLDA. Most synchronous CPUs require a minimum of three bus cycles between the time HOLD is activated and they can return HLDA. While the 82786 waits for a HLDA from the External Bus Master, the 82786 responds to slave accesses by the External Bus Master (see Section 4.3.2 "Priority Exceptions").

The 82786 does not incur overhead at the end of the transfer (HREQ inactive to HLDA inactive) because it can start a graphics memory access while it is relinquishing the bus to the External Bus Master. However, the External Bus Master does incur overhead.

4.4.1.2 RETAINING CONTROL OF THE SYSTEM BUS

The 82786 can require the bus for a lengthy period of time. If the Graphics Processor (GP) has been programmed with a high priority and the GP executes a command that requires a lot of access to system memory, the 82786 could hold the system bus for several consecutive accesses. For example, if the 82786 draws a long vector into a bitmap residing in system memory, this command requires a long bus access. The maximum time the 82786 can retain the system bus is determined by the frequency of graphics memory refresh cycles programmed by the Refresh Scaler value (see the calculations in Section 4.2.3 "DRAM/VRAM Refresh Control Register").

If the CPU must regain the bus before the 82786 completes the command, the CPU can remove HLDA early. The 82786 will complete its current access and remove HREQ to indicate to the CPU that it can take control of the bus. If the 82786 still requires more access to the system bus, it will reactivate HREQ after only two cycles. The CPU must recognize the request immediately to avoid a lockout condition in which the CPU is waiting for the 82786 to remove HREQ and the 82786 is waiting for the CPU to issue HLDA. This situation does not occur for synchronous interfaces (see Sections 4.5.1 "Synchronous 80286 Interface" and 4.5.2 "Synchronous 80186"). Extra logic may be required to extend the 82786 low time to prevent command lockout, if the 82786 is the External Bus Master in an asynchronous interface (see Section 4.5.3) and HLDA is ever removed prematurely; especially, if the CPU clock is significantly slower than the 82786 clock because the 82786 HREQ signal can drop faster than the CPU can acknowledge it.

4.4.2 Slave Interface

The 82786 Slave Interface allows the External Bus Master access to graphics memory or the 82786 Internal Registers and provides for synchronous operation with the 80186, 80286

and 80386 CPUs (see Sections 4.5.1 and 4.5.2). An asynchronous interface (see Section 4.5.3) also is supported for all other processors. The 82786 Slave Interface uses the following pins:

- 22 address inputs, A21:0.
- Byte High Enable Low ($\overline{BHE}$) input allowing byte access of 82786 memory.
- Bus command input signals Read Low ($\overline{RD}$), Write Low ($\overline{WR}$), and Memory or I/O Indication (MIO).
- Chip Select Low ($\overline{CS}$) input
- Slave Enable (SEN) output enables the 82786 data bus to connect to the external data bus. It is also a source of READY to the External Bus Master.

All these input signals, except $\overline{CS}$, are bidirectional pins driven by the 82786 when it executes in Master Mode (see Section 4.4.1.1 "Initiating Master Mode Interface"). When the 82786 is not in Master Mode, Slave Interface logic monitors these signals. The correct combination of bus commands generate a Slave cycle request, which the BIU arbitrates with graphics memory refresh, Display Processor (DP), and Graphics Processor (GP) requests.

4.4.2.1 INITIATING A SLAVE REQUEST

To initiate a Slave access (read/write), the External Bus Master asserts the Chip Select Low ($\overline{CS}$) input. The cycle can be directed at the 82786 memory or I/O mapped Internal Registers or graphics memory. The 82786 acknowledges the request by activating the Slave Enable (SEN) output, which indicates write data can be driven on the 82786 data pins, D15:0, or Read data is being output by the 82786 for an internal read or graphics memory access. SEN is also a source of READY for the External Bus Master that generated the slave cycle. The External Bus Master can wait a long time in a Ready loop, which can be minimized by programming the priority of the External request higher than other requests. For details, see Section 4.2.7 "External Priority Register" and Section 4.3 "Bus Arbitration."

After beginning a slave access to the 82786, the External Bus Master must enter a wait state because the 82786 does not queue slave requests and cannot respond to additional slave commands from the External Bus Master until the current slave access completes.

The 82786 indicates termination of the slave access by a high level on the SEN output. The data bus transceivers also can be enabled by SEN.

4.4.2.2 8-BIT AND 16-BIT INTERFACES

The 82786 can support either an 8-bit or 16-bit interface to access Internal Registers, based on the value of the BCP bit in the BIU Control Register (see Section 4.2.2). Following RESET, the BCP bit is set to 0, which indicates that the 82786 is in 8-bit Mode.

In 8-bit Mode, the Internal Registers must be accessed by two successive bus cycles. This is not necessary for reads, but is necessary for writes to the Internal Registers. The low byte (A0 = 0) must be written first, followed by the high byte (A0 = 1) of the Register. The

82786 latches the low byte in a temporary register until the high byte is written. The address bits A21:1 must be the same for both bus cycles. A register is not changed until the second byte (the high byte) is written. No restriction on the time between the two bus cycles exist, but if successive low bytes are written before a high byte is written, the last low byte is the one written to the Internal Register.

The 82786 has a lockout mechanism, which prevents a high byte write to modify an Internal Register if the temporary register does not contain a valid word. Both I/O and memory mapped 82786 Internal Registers operate in this way, which is consistent with the operation of 8-bit CPUs like the 80188 and the 8088. Word instructions in these processors execute by transferring the low byte first, followed by the high byte. A high byte write followed by a low byte write has no effect.

The 82786 does not cross bytes on output in 8-bit Mode: low bytes are transferred on the low data lines D7:0 and the high bytes on high data lines D15:8. An external crossover creates the 8-bit bus for the host. The external crossover is not additional hardware because the 8-bit host needs a crossover to access the memory array.

With an 8-bit processor, the following assembly routines can be used to load the 16-bit BIU Control Register with AX.

```
    mov   dx,BIUControl
    out   dx,al             ; write AL into low-byte of BIUControl
    mov   al,ah
    inc   dx
    out   dx,al             ; write AH into high-byte of BIUControl

or:

    mov   dx,BIUControl
    out   dx, ax            ; write AX into BIUControl word
```

To set the interface to 16-bit Mode and write the address of the Internal Register Block in the Relocation Register following RESET, access the BIU Control Register twice to write parameter values in the lower and upper bytes including a 1 for the BCP bit to indicate a 16-bit interface. With the BCP bit set to one, a 16-bit word access can be made to the Relocation Register to write the starting address of the Internal Registers in the lower and upper bytes. The following initialization code can be used.

```
    mov   dx,BIUControl
    mov   al,30h
    out   dx,al             ; write 30h into low-byte of BIUControl

    xor   al,al
    inc   dx
    out   dx,al             ; write 00h into high-byte of BIUControl

    mov   dx,InternalRelocation
    mov   ax,03F8h
    out   dx,ax             ; write 03F8h into InternalRelocation Word
```

In 16-bit Mode, the Internal Register Block is only word addressable. Odd word or odd byte accesses to internal locations will not produce desired results. Even byte access, however will work as desired. The least significant address bit, A0, is assumed to be 0 and is ignored in 16-bit Mode. Also, all the 82786 Master Mode operations are 16 bits wide, independent of the BCP bit, which means, in Master Mode, the system memory must be accessible 16 bits at a time.

4.4.2.3 SLAVE INTERFACE BYTE ACCESS TO MEMORY

The 82786 slave interface supports byte access to graphics memory. The combination of Byte High Enable Low ($\overline{BHE}$) and address line A0 generate the proper Write Enable Low Byte Low ($\overline{WEL}$) and Write Enable High Byte Low ($\overline{WEH}$) signals. $\overline{BHE}$ and A0 are ignored for read cycles. Slave cycles accessing graphics memory are the only times $\overline{WEL}$ might not immediately follow $\overline{WEH}$.

4.4.2.4 SLAVE ENABLE (SEN) AS READY INDICATION

Slave Enable (SEN) is optimized for connection to the 80286 Clock Generator and Ready Interface, the 82284 Asynchronous Ready (ARDY) or Synchronous Ready (SRDY) input, when the 82786 Slave Interface is set in Synchronous 80286 Interface (see Section 4.5.1 "Synchronous 82786/80286 System"). When operating in synchronization with the 80286 slave write cycles, the 82786 can always execute with a minimum of two wait states. The number of wait states for a read cycle is a function of the memory speed. For reads using two wait states, SEN is connected to SRDY. For reads using three wait states, it is connected to ARDY. Write cycles in both cases execute with two wait states because the 82786 issues SEN with different timing during write cycles.

4.4.2.5 ACCESSING INTERNAL REGISTERS & GRAPHICS MEMORY

The External Bus Master can access either 82786 memory or I/O mapped Internal Registers or graphics memory. The Internal Register address space consists of a contiguous 128-byte block (see Figure 1-3), which starts at an even-byte address and is mapped to memory or I/O space based on the state of the MIO bit in the BIU Relocation Register (see Section 4.2.1). The 82786 compares the value in the Internal Relocation Register with the incoming address to determine if the transfer is to graphics memory or memory or I/O mapped Internal Registers. Table 4-7 lists comparison possibilities and the corresponding action taken for each.

The 82786 ignores an I/O access if the 82786 does not have I/O mapped locations as the fifth entry in Table 4-7 indicates. The last entry illustrates self Chip Select logic. When the External Bus Master makes an I/O access to an 82786 that has I/O mapped Internal Registers, but the address of the request is not within the 128-byte range, the 82786 executes a dummy cycle without activating Write Low Byte Enable Low ($\overline{WEL}$) and Write High Byte Enable Low ($\overline{WEH}$) and does not generate Slave Enable (SEN). The dummy cycle wastes time and should be avoided by not generating the Chip Select Low ($\overline{CS}$) in these cases.

Table 4-8 displays the 82786 status or command interface for the 80286, which is identical to the decoding of the 82288 Bus Controller for the 80286.

Table 4-7. 82786 Address Comparison

Access Type	MIO Bit	Address Compare	Action Taken
Memory	mem	Not Equal	Memory Access
Memory	I/O	Not Relevant	Memory Access
Memory	mem	Equal	Internal Access
I/O	I/O	Equal	Internal Access
I/O*	mem	Not Relevant	—Ignored—
I/O*	I/O	Not Equal	—NOP Cycle—

* Can be masked out by the $\overline{CS}$ logic.

Table 4-8. 82786 Status Interface for the 80286

CPU $\overline{MIO}$ / $\overline{MIO}$	$\overline{S1}$ / $\overline{RD}$	$\overline{S0}$ / $\overline{WR}$	CPU Cycle Type	82786 Action
0	0	0	Intr. Ack	Ignored: No request generated
0	0	1	I/O Read	Internal Read (if addr =)
0	1	0	I/O Write	Internal Write (if addr =)
0	1	1	Idle	Ignored: No request generated
1	0	0	Halt/shutdown	Ignored: No request generated
1	0	1	Memory Read	Graphics/Internal Read (if addr =)
1	1	0	Memory Write	Graphics/Internal Write (if addr =)
1	1	1	Idle	Ignored: No request generated

4.4.2.6 COMMAND LOCKOUT

The 82786 does not queue commands from the External Bus Master. Most CPUs, like the 80286, issue status for one cycle and then hibernate until the cycle completes. The CPU should not issue another bus command until the 82786 responds to the initial request.

4.5 SYSTEM BUS INTERFACE

The 82786 can be easily connected to a variety of CPUs. The 82786 supports two Synchronous Interfaces, one for an 80286/80386 interface, the other for an 80186 interface, as well as an Asynchronous Interface for all other processors.

Both Master and Slave Interfaces can operate synchronously or asynchronously. In Master Mode, synchronous and asynchronous operation affects the sampling of the Hold Acknowledge (HLDA) signal only. In Slave Mode, synchronous and asynchronous operation also affect the sampling of Read Low ($\overline{RD}$) and Write Low ($\overline{WR}$) signals. Only the Read Low ($\overline{RD}$) and Write Low ($\overline{WR}$) inputs are synchronized to the system clock. Their values are sampled and latched prior to sampling the Memory or I/O (MIO), Chip Select Low ($\overline{CS}$), Byte High Enable Low ($\overline{BHE}$), and address A21:0 inputs, which must be valid with relation to the Read Low ($\overline{RD}$) and Write Low ($\overline{WR}$) pins.

The 82786 determines which mode to use based on the state of the ($\overline{BHE}$) and the MIO pins during RESET. Table 4-9 displays the interface modes defined by possible values of these pins.

As indicated in Table 4-9, synchronous operation is set if $\overline{BHE}$ is sensed high or MIO is sensed low at trailing RESET. Thus, either of these pins can be connected directly in 80286 synchronous systems because the $\overline{BHE}$ pin is driven high and the MIO pin is driven low during RESET. The 80186 tristates its $\overline{BHE}$ during RESET so a small static load on the signal can select asynchronous operation.

4.5.1 Synchronous 80286 Interface

The 82786 is optimized for the 80286, which explains the minimal 82786 interface logic. Figure 4-11 shows the 82786 connected synchronously to an 80286. Much of the logic, including the 82288 Bus Controller for the 80286, Chip Select, and READY, can be shared by the rest of the 80286 system.

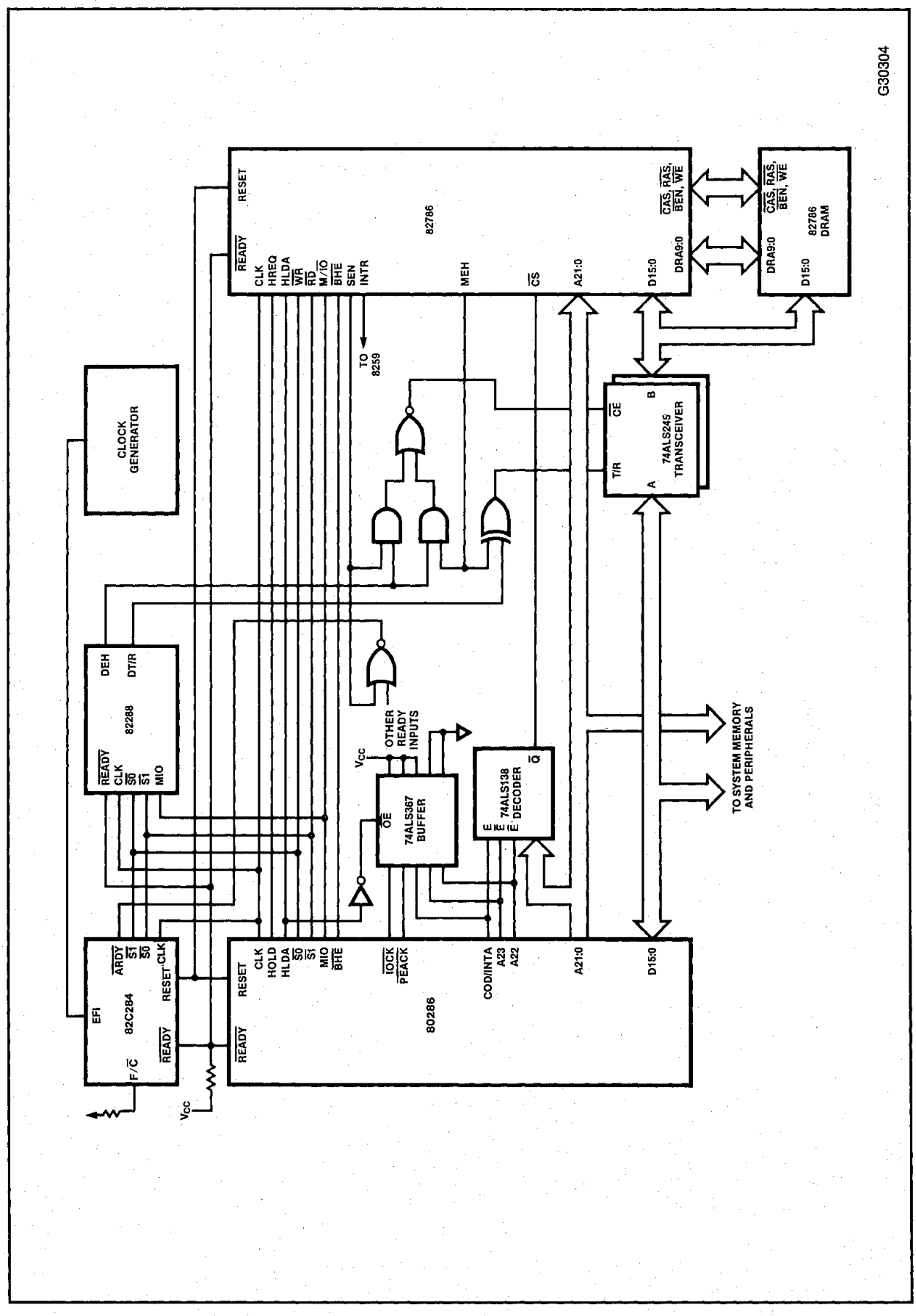
The 82786/80286 shows both master and slave accesses. The data transceivers allow the 80286 to access the 82786 Internal Registers and graphics memory; they also allow the 82786 to access the 80286 system memory. The data transceivers also provide the isolation required to allow the 80286 to access system memory while the 82786 simultaneously accesses graphics memory. The tristate buffer 74LS367 pulls the 80286 upper address lines, COD/INTA, LOCK Low, and PEACK to their proper states during Master Mode. If any of these signals are not used by the rest of the system, they need not be driven by the tristate buffer.

If Master Mode is not required, Master Enable (MEN), stays low and three of the four gates driving the data transceivers can be eliminated. Also, the tristate buffer, which is only used in Master Mode, can be eliminated. Hold Request (HREQ) should be left open and the 82786 Hold Acknowledge (HLDA) pin should be tied to ground so that the 82786 will never enter Master Mode.

Both the 80286 and the 82786 internally divide-by-two the Clk input and use both phases. For the 82786 to run correctly with the 80286, these phases must be correlated correctly. This can easily be done by observing the setup and hold times for rising RESET for both chips. The 82C284 Clock Generator and Ready Interface chip will meet the requirement.

Table 4-9. BHE# and MIO Pin Values

Operation Mode	BHE#	MIO
Synchronous 80286 bus	1	0
Synchronous 80186 bus	1	1
Asynchronous bus	0	X



ure 4-11. 82786 Synchronously Connected to an 80286

Depending on the CLK speed and the type of DRAM/VRAM used, the 82786 may have stringent Clk duty cycle requirements. Due to these requirements, the internal oscillator of the 82C284 Clock Generator and Ready Interface chip may not be sufficient, but it may be possible to use an external oscillator to drive the 82C284 External Clock (EFI) pin. To keep clock skew between the 80286 and the 82786 to a minimum, the chips should be placed as close together as possible.

When the 82786 bus is free, the circuit in Figure 4-11 permits CPU slave accesses using two wait states for writes and three wait states for reads.

With DRAMs capable of slightly faster DRAM access times, slave read and write accesses with two wait states can be achieved. The Synchronous Ready (SRDY) input is used instead of the Asynchronous Ready (ARDY). The 82786 Slave Enable (SEN) timing is such that a minimum of two wait states are always generated for writes, but a minimum of two or three wait states are used for read based on the use of SRDY and ARDY. With reads using two wait states, the SEN signal must be qualified with Chip Select Low ($\overline{CS}$) so SEN does not extend into the cycle following the slave write. The most critical relationship to be satisfied for writes using two wait states is:

$$Tcac, Tc + Tch - 15 - 45$$

For a 10 MHz 82786 the DRAM column access must be:

$$Tcac < 50 + 25 - 45 = 30 \text{ ns}$$

where:

Tcac = Cas Low Access Time
Tc = 82786 Clock Period
Tch = Clock High Time

For additional data regarding this relationship, refer to the *Intel 82786 CHMOS Graphics Coprocessor Data Sheet* and the *82786 Hardware Configuration Application Note.*

Also note that × 1 DRAMs have two transceiver delays. The critical timing calculations for the Slave Interface are calculated as shown in the example below of an 80286/82786 system running at 8 MHz.

```
Chip Select Logic = path from 80286 address to 82786 CS Pin

                  < 2 X Clock Period — Address Valid — Setup
                  < 2 X 286.T1       — 286.T13       — 82786.Ts1
                  < 2 X 50 ns        — 35 ns         — 5 ns
                  < 60 ns

Ready Logic       = path from 82786 SEN to 82284 SRDY Pin
 If ARDY used     < Clock Period     — SEN Active    — ARDY Setup
                  < 286.T1           — 82786.S18     — 82C284.T13
                  < 50 ns            — 25 ns         — 0 ns
                  < 25 ns
```

```
Ready Logic          = path from 82786 SEN to 82284 ARDY pin
 If SRDY used        < Clock Period      — SEN Active     — SRDY Setup
                     < 286.T1            — 82786.S18      — 82C284.T11
                     < 50 ns             — 25 ns          — 15 ns
                     < 10 ns

                                    from SEN active to read data valid
Read data valid ≥ 82786.Ts22 + transceiver delay

                                    from SEN active to write data valid
Write data valid ≥ 82786.Ts20
```

The Master Mode signals generated by the 82786 are all within the specification range guaranteed by the 80286, which means that system memory, designed to function with the 80286, also functions with the 82786. The only signals that may not be within the range of the 80286 specifications are the data bus signals due to transceiver delays. The memory subsystems accessed by the 82786 in Master Mode must meet the following requirements:

```
                        Data valid to falling clock after Tc phase 2
read data setup      > 82786 read data setup  +  transceiver delay
                     >     82786.T8           +  data in to data out
                     >       5 ns             +  Tprop

                        Data valid delay from falling clock after Ts phase 1
write data valid     < 82786 write data valid —  transceiver delay
                     <     82786.T14          —  data in to data out
                     <       40 ns            —  Tprop
```

The clock skew between the 80286 and the 82786 must be considered in all the above calculations.

4.5.2 80186 Synchronous Interface

The 82786 supports a synchronous status interface to the 80186. The 82786 and the 80186 must be driven with the same external clock (EFI). The Reset Inputs to the 82786 must be generated from $\overline{RES}$ for the 80186 by delaying it one (1) clock input. This guarantees that the 82786 Clock phase 1 is coincident with the 80186 CLKOUT Low. A synchronous 80186 interface is selected if BHE is high and MIO is high prior to falling RESET.

Generally, this 80186/82786 configuration uses a minimum of three wait states for the 82786 slave read and write accesses. Therefore, the WT bit in the 82786 BIU Control Register should be set (see Section 4.2.2). The 82786 slave accesses are initiated only when the 82786 Chip Select Low ($\overline{CS}$) is activated.

The 80186/82786 can use a minimum of two wait states, if the WT bit in the Control Register is set to 0 (see Section 4.2.2). Rather than waiting for the $\overline{CS}$ to go active, a slave access request can be initiated as soon as the 80186 status lines go active. If the 82786 is not in the midst of another bus cycle, and the 80186 request is the highest priority, the bus will immediately be granted to the 80186 and a bus cycle will be started. If the $\overline{CS}$ then goes active, the 82786 can complete the access within two wait states. If the $\overline{CS}$ does not go active because the 80186 is accessing its own memory, not the 82786 Internal Registers or graphics memory, then the 82786 aborts the bus cycle by running a dummy 82786 bus cycle.

If the system contains RAM or ROM in addition to the 82786 graphics memory, which the 80186 often accesses, using two wait states will probably hinder, rather than help performance. Each time the 80186 fetches from its own system memory, (such as an opcode fetch or operand access), and the 82786 bus is idle, the 82786 will waste time running a dummy cycle. Fortunately, the busier the 82786 is, the less likely it will be free when the 80186 initiates a bus cycle; and the 82786 is less likely to waste time running a dummy cycle.

4.5.3 Asynchronous Interface

Almost any CPU can interface to the 82786 with the asynchronous interface. The CPU and 82786 clocks are independent and can run at different clock speeds. For example, if the 80286 and the 82786 are connected asynchronously, but both processors run at approximately the same clock frequency, the minimum possible wait state is one more than for the corresponding synchronous interface.

Figure 4-12 shows a Slave, 10 MHz, 82786 interface to an 8 MHz 80186. At 10 MHz, the 82786 requires the address to be valid S17 = 80 ns after $\overline{RD}$ or $\overline{WR}$ falls. The address remains valid for S16 = 130 ns.

Because the 80186 address disappears during the same cycle $\overline{RD}$ and $\overline{WR}$ fall, the address must be latched. The address can be shared by other components on the 80186 bus. Due to indeterminate phase relationship between the CPU and the 82786 clocks, ensure the read and write data timings have enough slack. The CPU's Asynchronous Ready (ARDY) input determines when the read data is sampled and the write data is removed. The 82786 Slave Enable (SEN) signal also generates the READY signal to ensure the data is available. D-flip-flops can be used to delay SEN to delay the CPU ready signal. For a 10 MHz 82786, the following timing requirements exist:

```
                                From SEN active to read data valid
read data valid                       > 82786.Ts22 + Tprop
                                From SEN active to write data valid
write data valid                      > 82786.Ts20
```

To initially place the 82786 into an asynchronous interface mode, the 82786 Byte High Enable ($\overline{BHE}$) pin must be low during the falling edge of RESET. To ensure this, a latch for $\overline{BHE}$ such as the 74LS373 is tristated and an open-collector inverter pulls down $\overline{BHE}$ during RESET.

Although bus transfers of a synchronous bus are faster than for an asynchronous bus, an asynchronous interface can provide higher performance. For example, for a given display resolution, the Display Processor overhead of a 10 MHz 82786 is a lower percentage of the total bus throughput than for an 8 MHz 82786. If the 82786 is used with a 16 MHz 80386, the 10 MHz 82786 would have more bandwidth for the CPU and Graphics Processor than the synchronous 8 MHz 82786. Generally, this means the CPU access will be completed faster with the asynchronous interface, provided the host clock is significantly slower than a 6 MHz 82786 system clock as with the 80286 and a 10 MHz 82786.

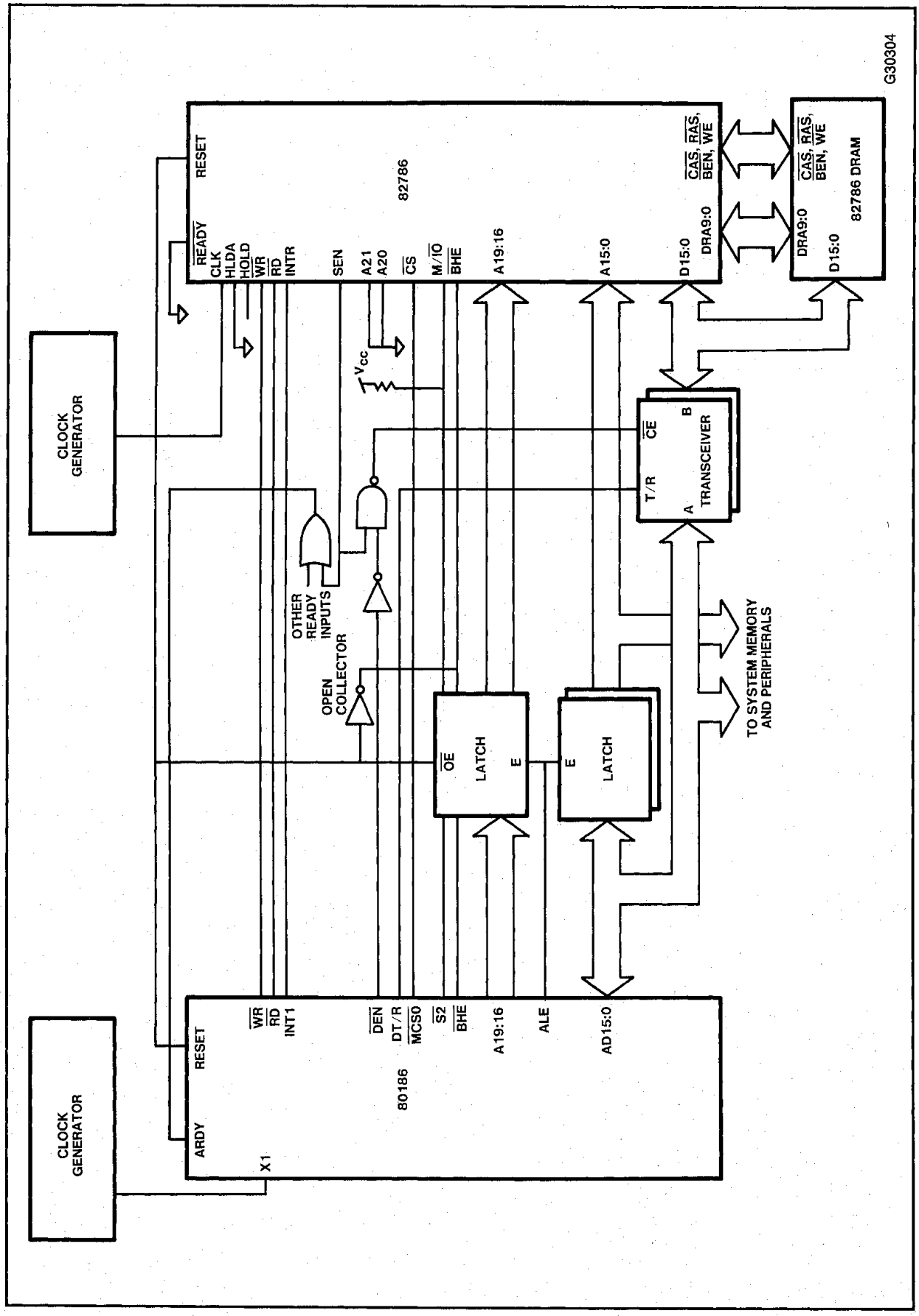


Figure 4-12. Asynchronous Slave 10 MHz 82786 Interface to 8 MHz 80186

4.6 PERFORMANCE

Slave performance is measured by assuming a request is made to an idle 82786. Fast memory refers to DRAMs/VRAMs with row access (Trac) time less than 120 nanoseconds and column access time (Tcac) below 45 nanoseconds. A synchronous interface is assumed. The minimum 80286/80386 wait states for fast memory is two. The minimum 80286/80386 wait states for slow memory is three. These values are for read cycles. In some cases, write cycles can operate with fewer wait states. For instance, the 80286 can always execute synchronous write cycles with two wait states (see Sections 4.4.2.4 "Slave Enable (SEN) as Ready Indication," 4.5.1 "Synchronous 82786/80286 System").

For asynchronous interfaces, if the host CPU is operating at the same frequency as the 82786, the number of wait states are typically one more than those indicated above. For CPUs operating at a slower frequency than the 82786, the number of wait states are, on the average, less than one greater than those previously given. However, an asynchronous interface, such as a 6 MHz 80286, can give less wait states than those previously quoted for the synchronous interface because the host clock is significantly slower than the 82786 10 MHz system clock.

4.7 RESET CONDITIONS

When the 82786 is RESET, several special pins are sampled and their state determines operating modes. RESET also effects all Internal Registers, which must be reinitialized following a RESET. The following subsections discuss the 82786 pins affected during RESET and initialization events.

4.7.1 Special Pins

The 82786 uses the Byte High Enable ($\overline{BHE}$), Read Low ($\overline{RD}$), and Write Low ($\overline{WR}$) user pins to set states at the trailing edge of RESET.

The $\overline{BHE}$ pin determines whether the $\overline{RD}$, $\overline{WR}$, and Hold Acknowledge (HLDA) pins are treated asynchronously or synchronously and establishes the command signal timings for the Slave Interface. If $\overline{BHE}$ is high, the 82786 sets synchronous operation; if $\overline{BHE}$ is low, the 82786 sets asynchronous operation. This is optimized for connection to the 80286 $\overline{BHE}$ output in synchronous operation, which also is high following RESET.

The $\overline{RD}$ and $\overline{WR}$ pins, if active (LOW) on trailing RESET, are used to set 82786 test modes, described in Appendix A. These pins should only be held active a maximum of 1/2 system clock after trailing RESET to avoid the 82786 detecting a Slave Interface Request.

4.7.2 Initialization

The BIU Internal Registers are set to default values on RESET as defined in the Internal Register descriptions in Sections 4.2.1 through 4.2.7. The BIU is ready to accept requests for bus cycles after trailing RESET. Three graphics memory refresh requests automatically occur following trailing RESET. These graphics memory refresh requests are the first tasks

that the 82786 performs after RESET. All RASes go active. The graphics memory refresh address is random unless the special DRAM/VRAM Test Mode was selected with the proper combination of $\overline{RD}$, $\overline{WR}$, and MIO at trailing RESET (see Appendix A "Test Modes"). The Graphics Processor (GP) and Display Processor (DP) will not issue any requests until the External Bus Master generates Graphics Processor Command Blocks or Display Control Blocks instructing the processors to perform tasks.

The 82786 does not have a direct warm reset capability, but DRAM/VRAM integrity can be maintained through a short duration RESET by guaranteeing refresh. The refresh request generator continues to count throughout RESET; the refresh address counter is not reset and refresh requests are queued three deep, which allows the RESET pulse to be quite long before violating DRAM/VRAM refresh requirements (see Section 4.2.3 "DRAM/VRAM Refresh Control Register"). All Internal Registers are reset, when the 82786 is RESET so they must be reconfigured each time a "warm RESET" is issued.

Graphics Memory 5

CHAPTER 5
GRAPHICS MEMORY

The 82786 supports up to 32 DRAMs or VRAMs without additional external logic. This support allows the 82786 to use cost effective memory devices with improved performance by using standard Page Mode, Fast Page Mode, Static Column Decode, or Ripplemode DRAMs. The Fast Page Mode, Static Column Decode, and Ripplemode memory devices enable the 82786 to cycle the DRAMs in the 100 ns range rather than the 200 ns range required by most Page Mode devices.

5.1 DRAM/VRAM CONFIGURATIONS

The 82786 supports a range of DRAM/VRAM configurations, which can include:

- Interleaving or noninterleaving (1 or 2 banks)
- Single or mutliple rows per bank (1, 2, 3, or 4)
- Width (×1, ×4, or ×8)
- Height (16K, 64K, 256K, or 1M)
- Performance (Page Mode or Fast Page Mode)

The only limitation is 4 MBs of address space. The 82786 DRAM/VRAM address lines (DRA9:0) can drive 32 memory devices directly, while the $\overline{RAS}$, $\overline{CAS}$, $\overline{WE}$, and $\overline{BEN}$ lines can drive 16 devices directly. When the memory array consists of more than 32 interleaved or 16 noninterleaved devices, external drivers must be used to drive the memory array.

Eight possible memory array structures can exist using 1 to 4 rows of interleaved or noninterleaved memory. Any of the 4-bit (×4) or 8-bit (×8) devices can be implemented in any of the eight possible configurations without exceeding the number of devices or the 4 MB address space limit. Figures 5-1 and 5-2 map the connections for ×4 and ×8 devices.

The configurations are standard. The two $\overline{CAS}$ signals delineate the two interleaved banks in Figure 5-2, the ×8 example. Bank 0 contains even words; Bank 1 contains odd words. Each of the four $\overline{RAS}$ signals drives a unique row in one or both banks based on the memory configuration. The DRAM/VRAM address signals, DRA9:0, are not shown because they must always drive the entire array. Similarly, the Write Enable Lines ($\overline{WE}$s) are not shown because each line must drive every low or high byte in the array.

Using 256K devices, Figure 5-3 provides the correlation of the address with the $\overline{RAS}$ and $\overline{CAS}$ lines to illustrate how to determine the location of bad memory during testing. The BIU DRAM/VRAM Control Register defines the memory configuration; refer to Section 4.2.4 for details on this register.

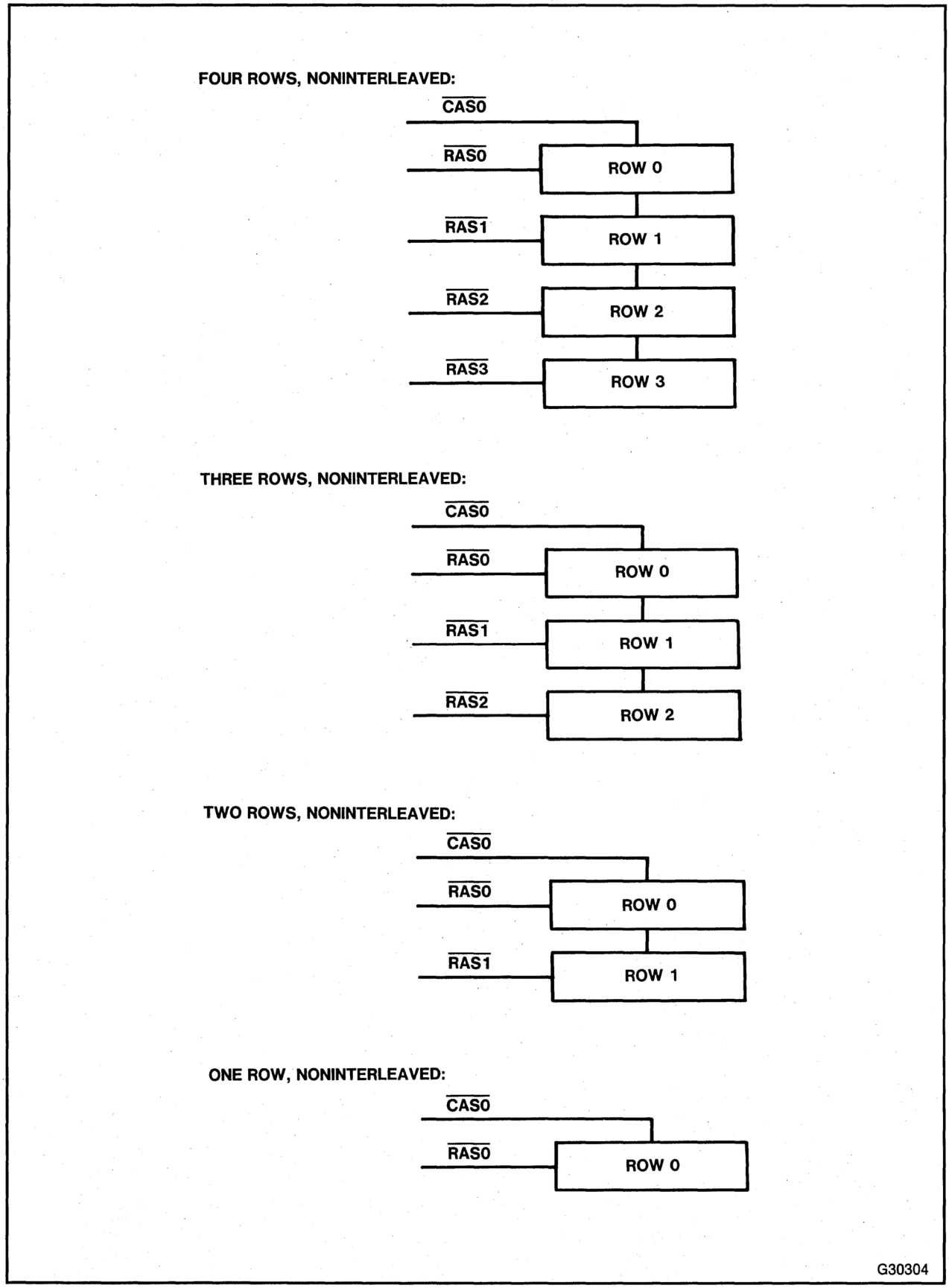


Figure 5-1. Noninterleaved DRAM Arrays

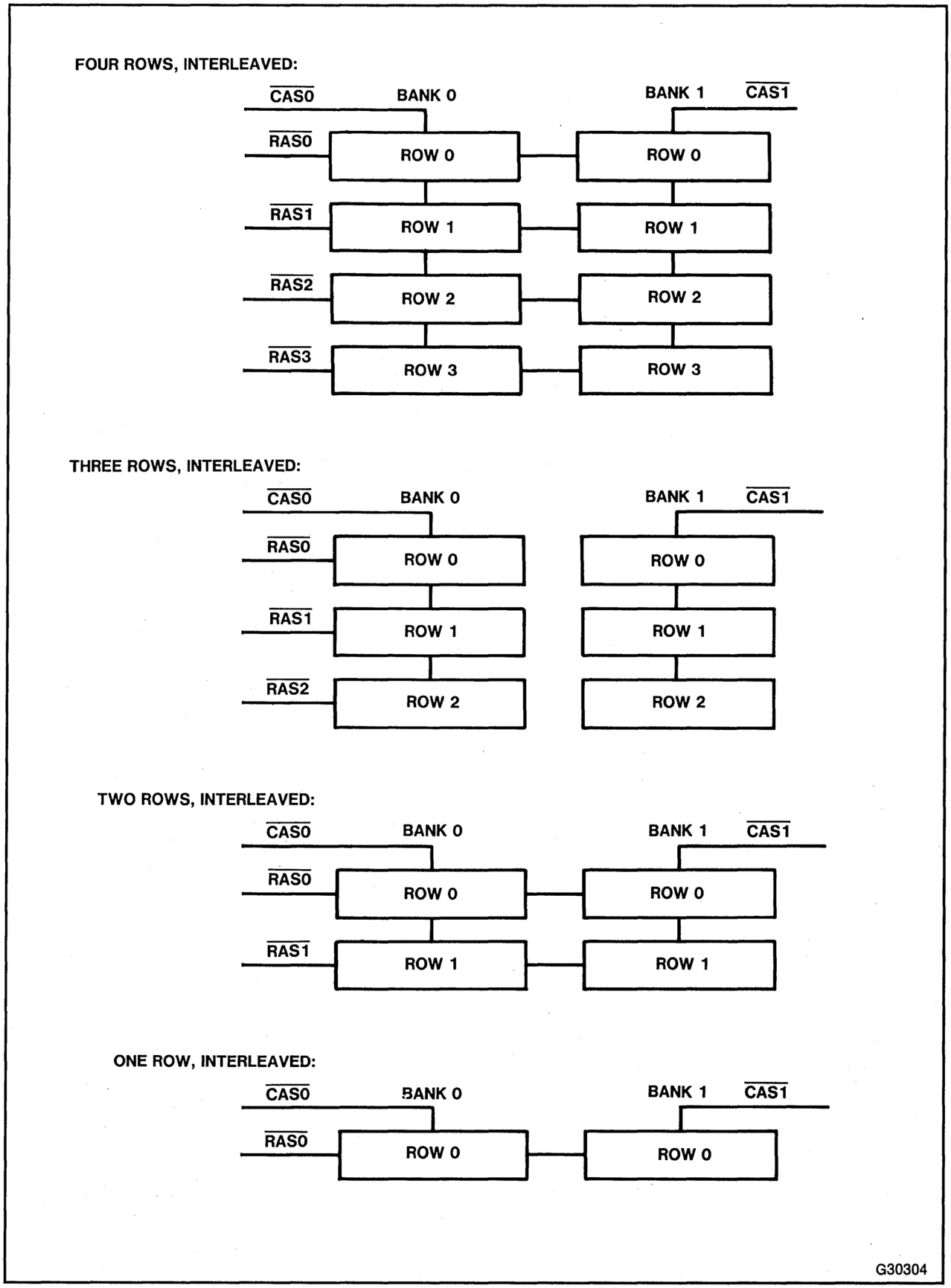


Figure 5-2. DRAM Arrays with ×8 Devices

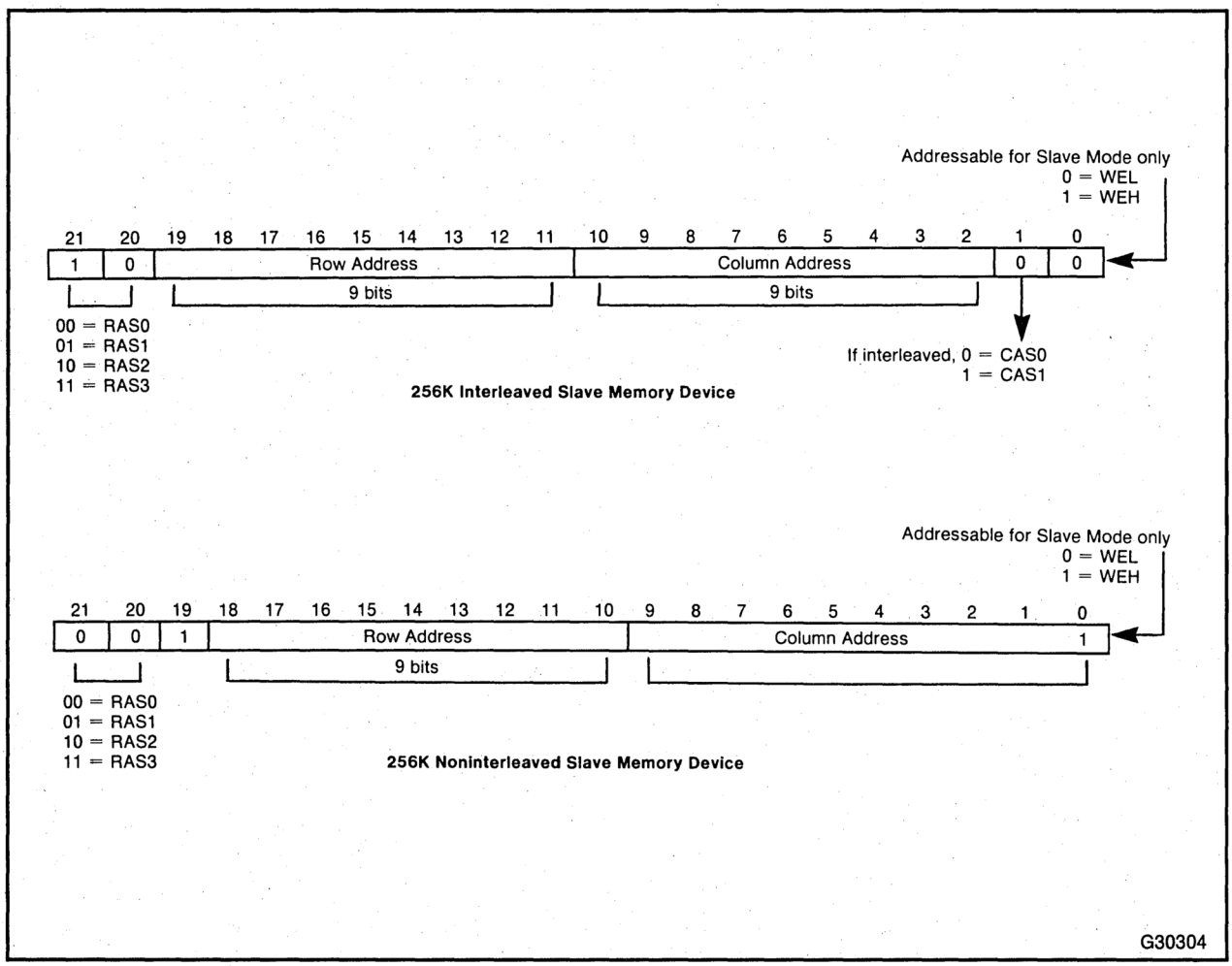


Figure 5-3. Correlation of Address with $\overline{RAS}$ and $\overline{CAS}$

5.1.1 VRAM Considerations

The 82786 supports a VRAM Mode for using dual port video DRAMs (VRAMs) to generate the video data stream. In VRAM Mode, the bits per pixel (bpp) are fixed in external hardware. All Tile Descriptor parameters and Display Processor Registers referring to video data fetch should be programmed to zero (see Sections 3.1.3.2 "Strip Descriptor Format" and 3.3.2 "Display Control Block Registers") unless the VRAM/DRAM overlay Mode is used (see Section 3.2.5.2 "Hardware Overlays"). With hardware overlays, only the first Tile Descriptor in the strip is programmed for VRAM Mode. Tile Descriptors for the second tile and any subsequent tiles in the strip follow DRAM programming requirements. If more than two tiles do not exist and the second tile is smaller than the strip display area, the remaining area is defined as Field by the F bit in the Tile Descriptor.

To enable VRAM Mode, set the VR bit in the BIU Control Register (see Section 4.2.2) to 1. For the first tile in each Strip Descriptor, a VRAM Data Transfer (DT) cycle occurs. The VRAM Shift Register is loaded from the selected memory row. The Shift Clock (SClk) to the VRAMs is clocked by the Video Clock (VClk) frequency and enabled when Blank is low. Then, the data is serialized by a counter controlled by the dot clock frequency. The DT cycle occurs during Blanking; the SClk is activated during display time. Normal DRAM cycles occur for any subsequent tiles in each strip. No limit exists on the number of strips supported. VRAMs shift 256 words each transfer so the pixel data for every scan line in the entire display must be contained in a single row in memory (256 words for noninterleaved; 512 words for interleaved memories). The Tile Descriptor for the first tile in the strip must indicate only one pixel. The address specified for this pixel corresponds to the first pixel displayed. For details, see Table 3-1 in Section 3.1.3.2 "Strip Descriptor Format" and 3.2.5 "VRAM Support."

During the horizontal retrace period, the 82786 transfers the contents of the memory row containing the first pixel into the VRAM shift register. The VRAM shift clock is gated with a Blank signal. During the active display time, the shift clock is active and periodically clocks out the video data. External multiplexers must be used to convert the 16-bit (32-bit, if interleaved) data stream into a serial stream to obtain the bpp needed as shown in Figure 5-4.

5.1.2 Considerations for ×1 Memory Devices

Use of ×1 devices in some configurations require special treatment by the 82786. The separate Data-In (DIN) and Data-Out (DOUT) pins of the ×1 components require a tristate buffer for the DOUT lines of each bank. All rows within the same bank can share the same tristate buffer. Figure 5-5 displays a full connection diagram consisting of 32 256K ×1 DRAMs including the tristate buffers.

The two interleaved banks, each consisting of one row, receive special treatment from the $\overline{\text{RAS}}$ lines. Normally, $\overline{\text{RAS0}}$ drives all the devices in both banks for the row. However, the $\overline{\text{RAS}}$ lines can drive only 16 devices and the configuration contains 32. The 82786 recognizes this special case, and because $\overline{\text{RAS1}}$ is not being used, the 82786 automatically drives $\overline{\text{RAS1}}$ identically to $\overline{\text{RAS0}}$ as outlined in Example 1.

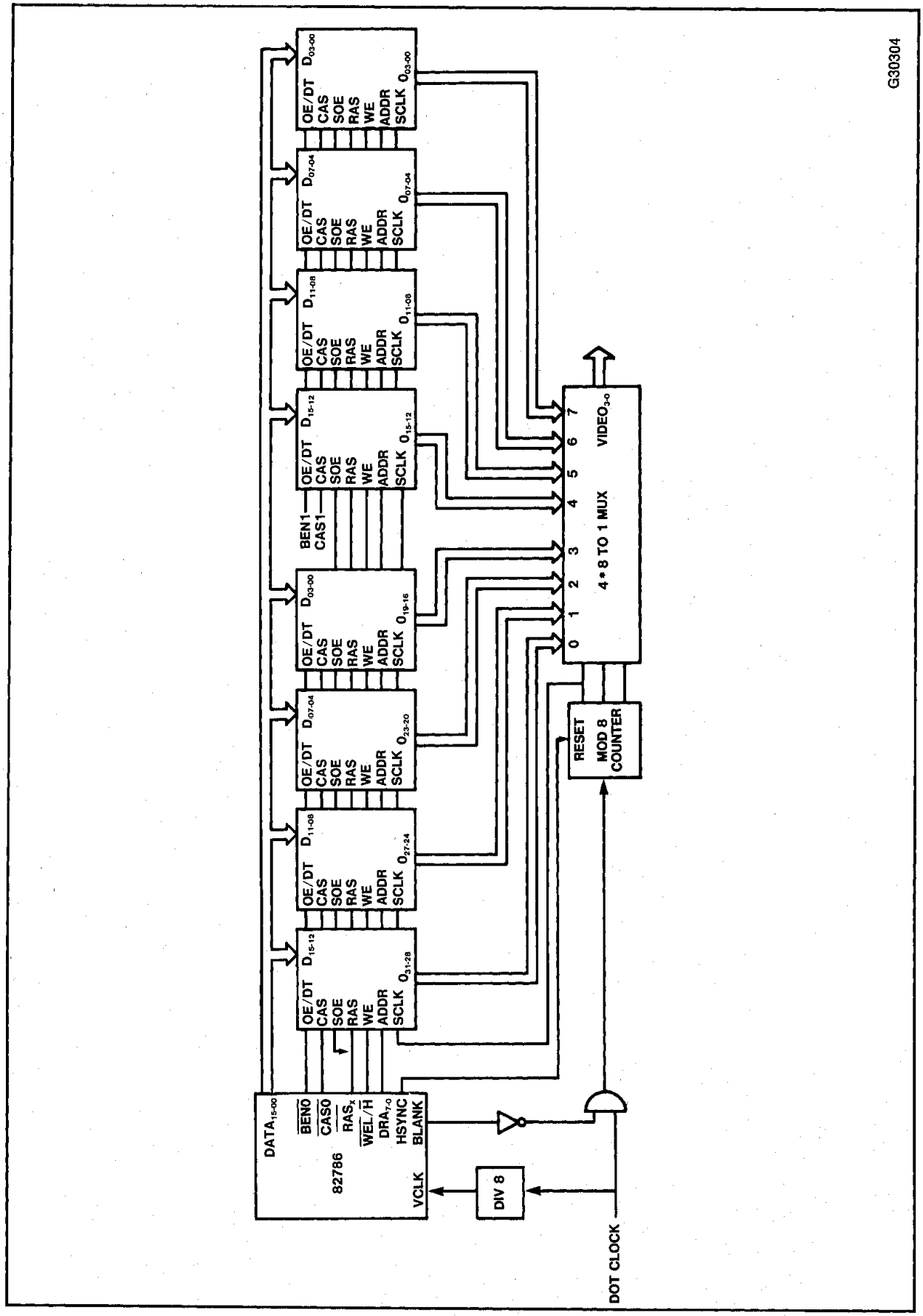
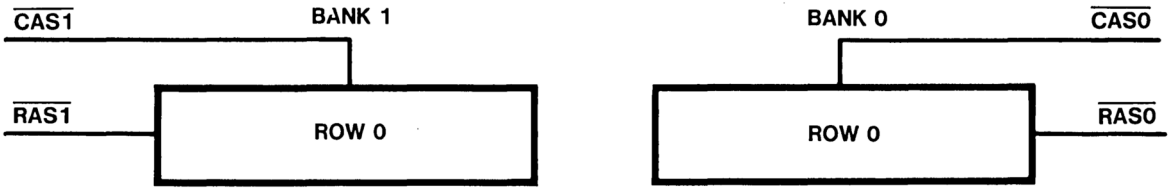


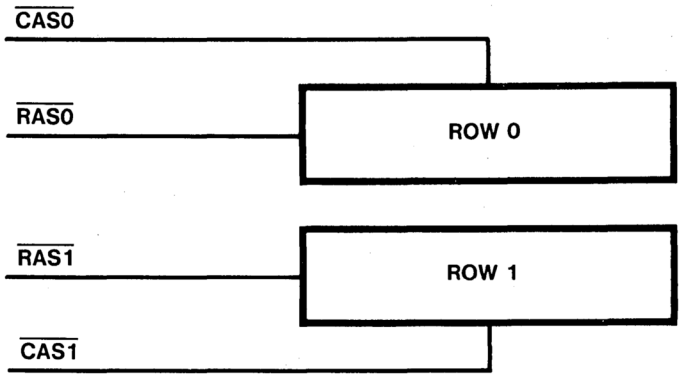
Figure 5-4. External Multiplexers Provide 4 Bpp

Example 1: One Row of 32 Interleaved ×1 Devices



Example 2 also illustrates a special ×1 DRAM case, in which the 82786 drives two rows of noninterleaved ×1 DRAMs. This configuration requires only one bank of tranceivers, but block transfer time is reduced by half of that obtained in Example 1. Normally, $\overline{CAS0}$ drives all 32 DRAMs, but due to drive limitations, both $\overline{CAS0}$ and $\overline{CAS1}$ are used (one for each bank). The 82786 recognizes this situation and drives $\overline{CAS1}$ identically to $\overline{CAS0}$ as outlined in Example 2. The 82786 also does this with similar ×4 and ×8 DRAM configurations.

Example 2: Two Rows, Each with 16 Noninterleaved ×1 Devices



The third special case occurs with 1 M ×1 DRAMs. The 82786 multiplexes two mutually exclusive signals, DRA9 and $\overline{RAS3}$, on the same pin. DRA9 is needed for 1M ×1 DRAMs only. Configuring more than two rows is impossible due to address space limitations, thus $\overline{RAS3}$ is not needed. When 1 M height is programmed in the DRAM/VRAM Control Register (see Section 4.2.4), the DRA9/$\overline{RAS3}$ pin functions as DRA9; otherwise, it functions as $\overline{RAS3}$.

5.1.3 Illegal DRAM/VRAM Control Register Values

The 82786 does not protect against programming the DRAM/VRAM Control Register with illegal combinations. For example, it can configure memory to contain 1 M × 4 devices in four interleaved rows, which specifies 16 MBs of memory. Results are unpredictable if such values are selected. All parameters in the DRAM/VRAM Control Register should be written at the same time to avoid producing illegal combinations.

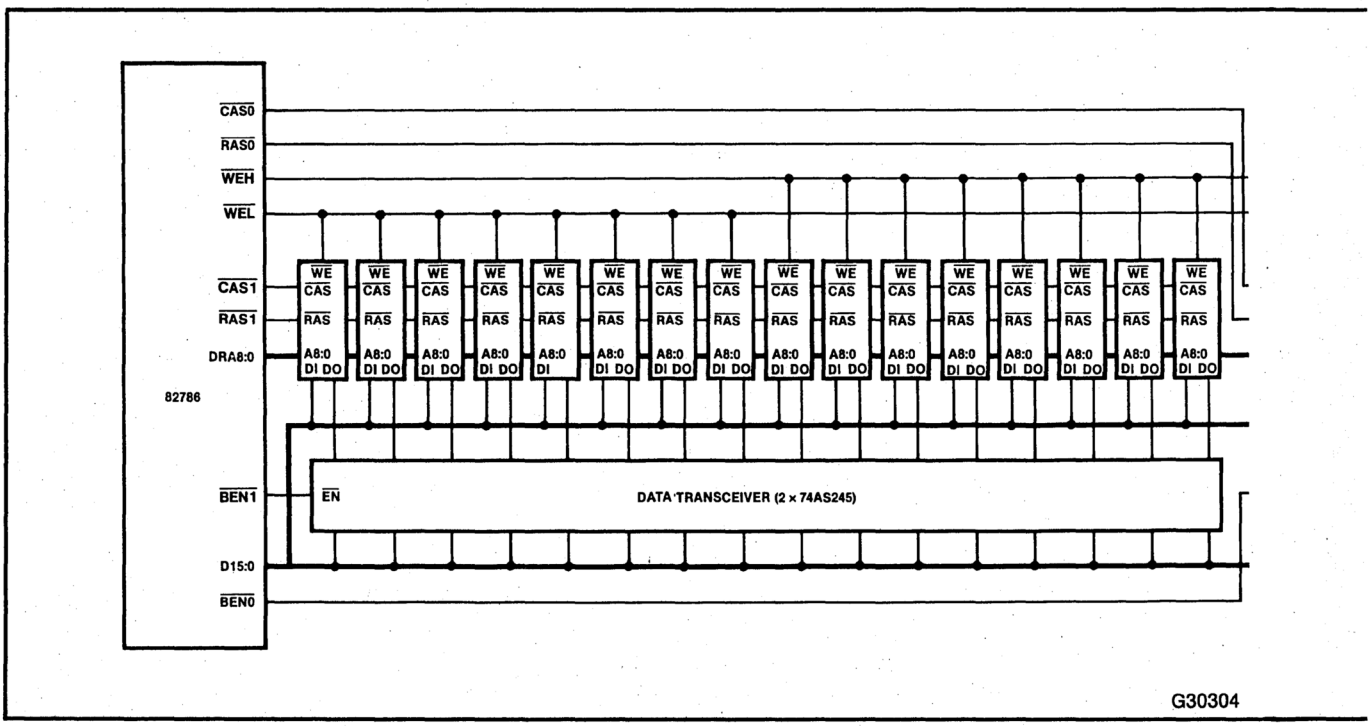


Figure 5-5. Two Interleaved Banks of 256K × 1

Table 5-1 shows memory configurations that can be used with the 82786 for 64K, 256K, and 1 M DRAMs/VRAMs. The top number in each box indicates the total memory size in bytes; the bottom number is the number of devices required.

5.1.4 Data Line Connection

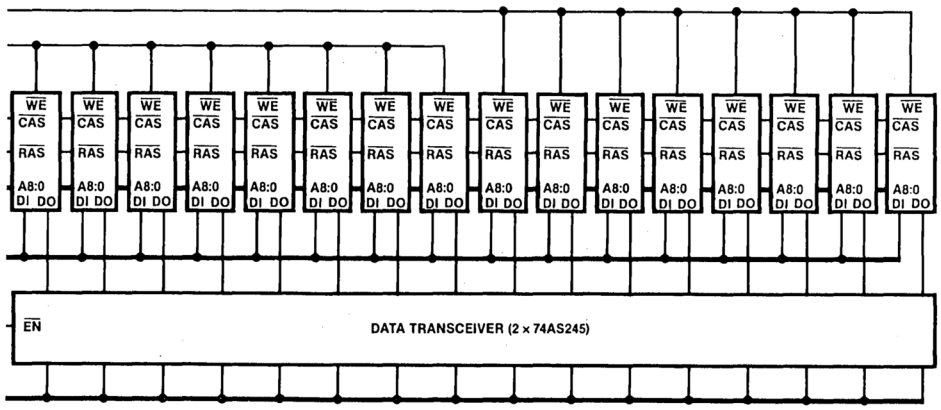
The data line connection varies based on the organization of the memory device I/O lines. The ×1 devices have unique Data-In (DIN) and Data-Out (DOUT) lines while ×4 and ×8 DRAMs have common I/O. The 82786 issues two signals, Bank Enable 0 ($\overline{BEN0}$) and Bank Enable 1 ($\overline{BEN1}$), to enable output to the memory array. $\overline{BEN0}$ enables Bank 0 to drive the data bus (D15:0), and $\overline{BEN1}$ enables Bank 1. The ×1 devices require a transceiver between the DRAM DOUT pins and the 82786 D15:0 lines as shown in Figure 5-6.

Common I/O devices have output enables incorporated in them so the BENs are connected directly to the Output Enable (OE) pin as Figure 5-7 illustrates.

5.2 DRAM/VRAM CYCLE TYPES

The 82786 supports two cycle types: single and block. A single cycle operates on one 16-bit word; a block transfers a minimum of two 16-bit words but does not restrict maximum length. Table 5-2 displays supported single cycle types and their cycle times. The cycle times are counted in system clocks (Clks), which is 1/2 the CLK input frequency.

Supported block cycle timings vary based on the memory device type and configuration as outlined in Table 5-3.



G30304

Figure 5-5. Two Interleaved Banks of 256K × 1

Table 5-1. Memory Configurations

	Noninterleaved				Interleaved			
	1-row	**2-rows**	**3-rows**	**4-rows**	**1-row**	**2-rows**	**3-rows**	**4-rows**
64K x1	128K 16	256K 32	384K 48*	512K 64*	256K 32	512K 64*	768K 96*	1024K 128*
16K x4	32K 4	64K 8	96K 12	128K 16	64K 8	128K 16	192K 24	256K 32
8K x8	16K 2	32K 4	48K 6	64K 8	32K 4	64K 8	96K 12	128K 16
256K x 1	512K 16	1024K 32	1536K 48*	2048K 64*	1024K 32	2048K 64*	3072K 96*	4096K 128*
64K x4	128K 4	256K 8	384K 12	512K 16	256K 8	512K 16	768K 24	1M 32
32K x8	64K 2	128K 4	192K 6	256K 8	128K 4	256K 8	384K 12	512K 16
1M x1	2M 16	4M 32	—	—	4M 32	—	—	—
256K x4	512K 4	1M 8	1.5M 12	2M 16	1M 8	2M 16	3M 24	4M 32
128K x8	256K 2	512K 4	768K 6	1M 8	512K 4	1M 8	1.5M 12	2M 16

*Requires external buffering

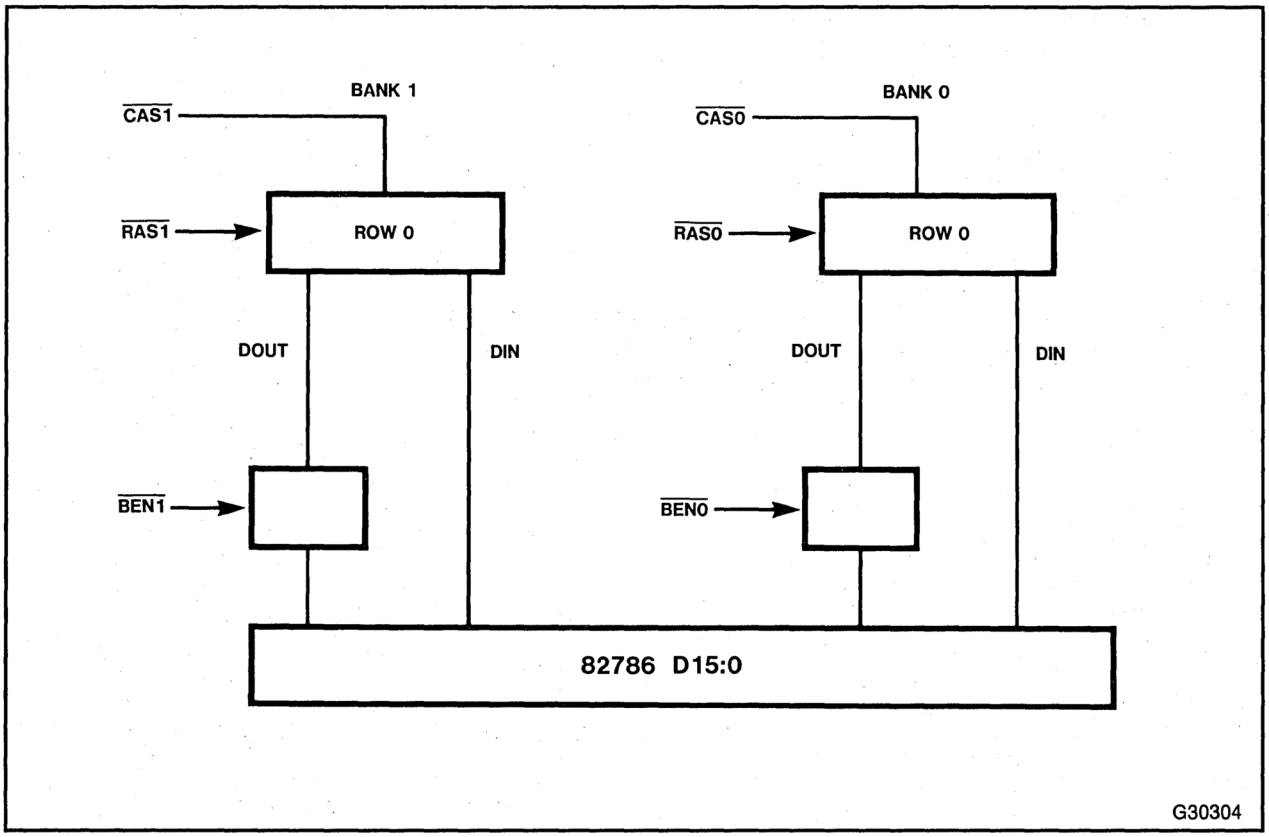


Figure 5-6. Data Line Connections ×1

5.3 GRAPHICS MEMORY REFRESH

The BIU handles graphics memory refresh. The BIU Refresh Control Register contains a programmable Refresh Scalar, which sets the refresh rate. The Refresh Scalar value is repeatedly loaded into a counter and decremented every 16 input clocks, which is every 800 ns at 10 MHz. This allows for 800 ns granularity on the refresh period. The default value of 18 in the Refresh Scalar at RESET establishes a 15.2 μS period between refresh requests, if the 82786 is operating at 10 MHz, which results in a single row (1 of 128) refresh request being issued every 1.9456 mS. For more details, refer to the calculations in Section 4.2.3 "DRAM/VRAM Refresh Control Register."

5.3.1 Refresh Latency

Latency exists between the graphics memory refresh request and the graphics memory refresh cycle. The latency is predictable except when the 82786 is in a READY loop while executing a bus cycle on the External Bus because the number of wait states encountered cannot be predicted. The longest latency occurs when a graphics memory refresh request becomes active just after the 82786 receives a Hold Acknowledge (HLDA) from the host CPU to execute a block transfer on the External Bus. Graphics memory refresh requests can interrupt the block transfer, but only on a doubleword boundary. The 82786 must execute two full bus cycles on the External Bus before the graphics memory refresh cycle executes.

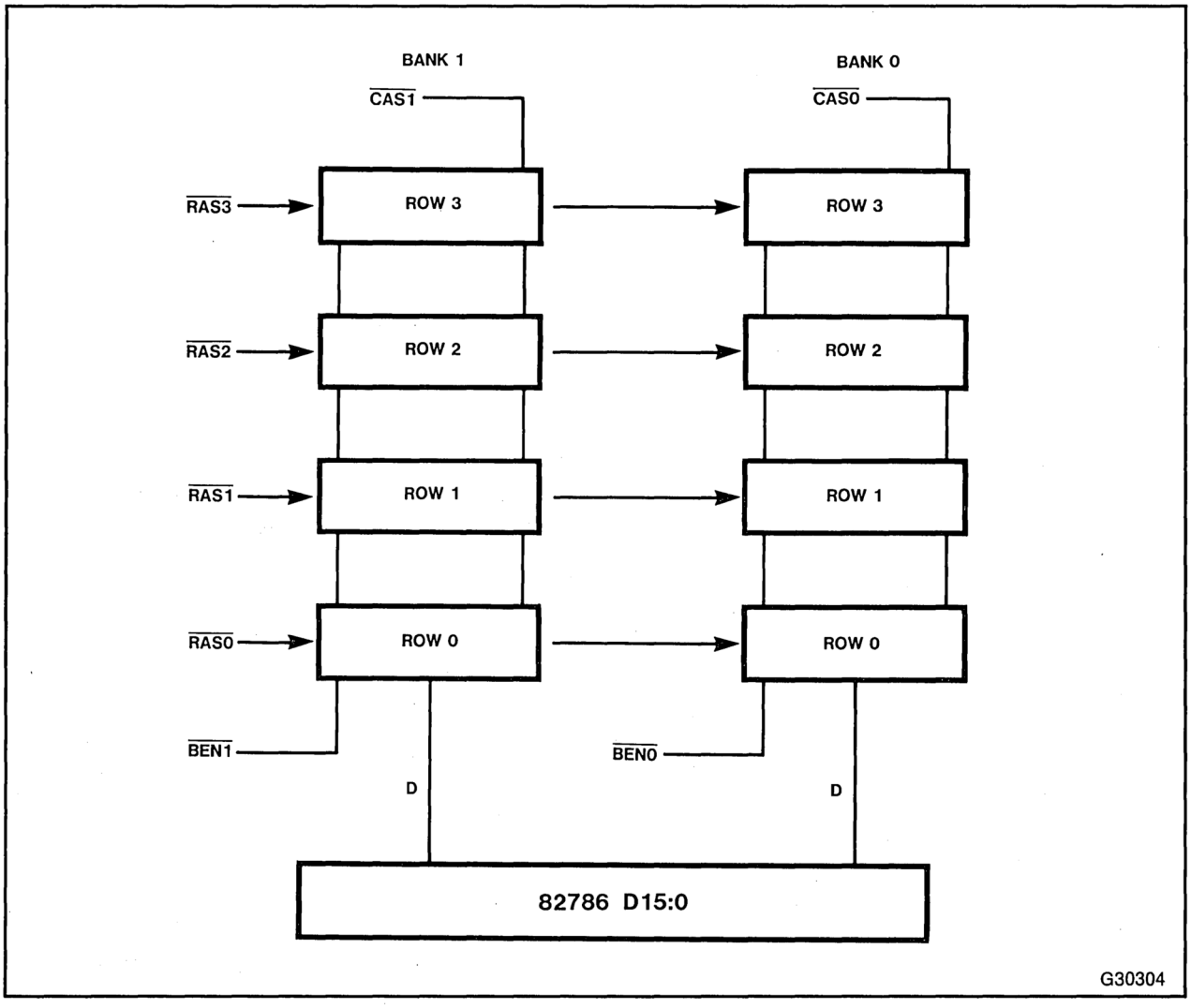


Figure 5-7. Data Line Connections ×4 and ×8

Table 5-2. DRAM Single Cycle Times

Cycle Type	Cycles	Times
Single Reads	3	300 ns @ 10 MHz
Single Writes	3	300 ns @ 10 MHz
Read-Modify-Writes	4	400 ns @ 10 MHz

Table 5-3. DRAM Block Transfer Rates

Mode	Cycles	Rate
Page, Noninterleaved	2	10 Mb/s @ 10 MHz
Page, Interleaved	1	20 Mb/s @ 10 MHz
Fast Page, Noninterleaved	1	20 Mb/s @ 10 MHz
Fast Page, Interleaved	.5	40 Mb/s @ 10 MHz

The possibility of many wait states while these two bus cycles execute creates a need for a large graphics memory refresh latency tolerance. The 82786 handles this with a latency tolerance that is adequate for most system designs. With the default Refresh Scaler value outlined in Section 5.3 and DRAMs that require refresh every 2 mS, a 54.4 μS (2.0 mS − 1.9456 mS) latency tolerance exists before the graphics memory refresh latency is violated for any one row. However, the next graphics memory refresh request occurs 15.2 μS into this period. By the time this maximum tolerable latency has occurred, three additional graphics memory refresh requests have been made at 15.2, 30.4, and 45.6 μS into the 54.4 μS period. To avoid losing these graphics memory refresh requests and still maintain maximum latency, the 82786 queues as many as three refresh requests. This allows for a maximum latency of 45.6 μS between the graphics memory refresh request and the refresh cycle before a refresh request is lost. After a long latency is incurred ($>$15.2 μS), the queued graphics memory refresh requests have the highest priority of any request (see Section 4.3.2 "Priority Exceptions") and cause refresh cycles to occur in succession. Also, to ensure the maximum latency is not exceeded, each bus cycle to external system memory should not have more than 225 wait states. For additional details on latency, refer to the calculations in Section 4.2.3 "DRAM/VRAM Refresh Control Register."

5.3.2 Refresh after RESET

Following RESET, three graphics memory refresh cycles are executed first. The Refresh Scalar is loaded into the counter on the trailing edge of RESET.

A graphics memory refresh cycle issues all RASes at once. For configurations of less than four rows, only those RASes used are issued. No other DRAM signals go active. A graphics memory refresh cycle is a RAS-only refresh cycle, which has a cycle time of 3 system clocks or 300 ns at 10 MHz and is identical to a single read or write cycle.

The graphics memory refresh address is a 10-bit quantity, which is incremented by one each refresh cycle. Following a normal RESET, the value of the refresh address is indeterminate. The refresh address is set to a predetermined value only if the 82786 is brought out of RESET in BIU Test Mode as described in Appendix A. Not modifying the refresh address on RESET allows for a warm RESET: the contents of memory remain valid only if RESET is short enough that it does not violate the maximum refresh latency (see the calculations in Section 4.2.3 "DRAM/VRAM Control Register." Graphics memory refresh continues at the proper row after RESET goes inactive again.

Accesses to the 82786 Internal Registers, when they are memory mapped (see Section 4.2.1 "Relocation Register"), result in refresh-like cycles on the DRAM causing a RAS, but no other DRAM signal to go active.

5.4 CONFIGURING AND ACCESSING GRAPHICS MEMORY

The BIU DRAM/VRAM Control Register (see Section 4.2.3) configures graphics memory. The default value of this BIU Internal Register configures the graphics memory array to consist of 4 rows of noninterleaved Page Mode 256K $\times$1 devices. Graphics memory accesses should not be attempted until the memory array is configured properly and refreshed.

The 82786 has no memory initialization logic. Many memories require a minimum number of RAS cycles before they perform according to specifications. The system must either wait for refresh cycles to execute the required number of cycles, or the boot software on the host can quickly access the memory array for the required number of cycles.

5.5 MEMORY MAP

The memory maps for the 82786 and the CPU can differ. A memory map from the viewpoint of the 82786 is shown in Figure 5-8. The Graphics Processor (GP) and Display Processor (DP) can access graphics memory using a 22-bit address which allows access to the full 4 MB address range of the 82786. Both processors can access memory only; neither can make I/O accesses.

The 82786 dedicated graphics memory always starts at 000000h and ascends upward. The upper address varies based on the amount of memory configured in the BIU DRAM/VRAM Control Register (see Section 4.2.4). System memory begins at the top, graphics memory address + 1 and can ascend to the highest addressable memory location 3FFFFFh.

A memory map displaying the CPU's perspective is shown in Figure 5-9. The area that the 82786 graphics memory is mapped into can be anywhere in the CPU address space, and is defined completely by the address decode logic of the CPU system. Normally, only the space for the configured graphics memory is mapped into the CPU address space. If addresses above the configured graphics memory address space are mapped into the CPU address space and the CPU writes to addresses above the configured 82786 memory, the write will be ignored. If it reads from these locations, the data returned is undefined.

The 82786 Internal Registers can be configured to reside in memory or I/O space. If configured to reside in memory, then they will override the 128-byte area of the 82786 memory space for external (CPU) accesses. The Internal Registers are only accessible by the external CPU and therefore are never found in the 82786 GP or DP memory maps.

If the 82786 is configured with 1 MB of graphics memory and is used in an 80286 system, the memory maps and connection diagram might resemble those in Figure 5-10. All the

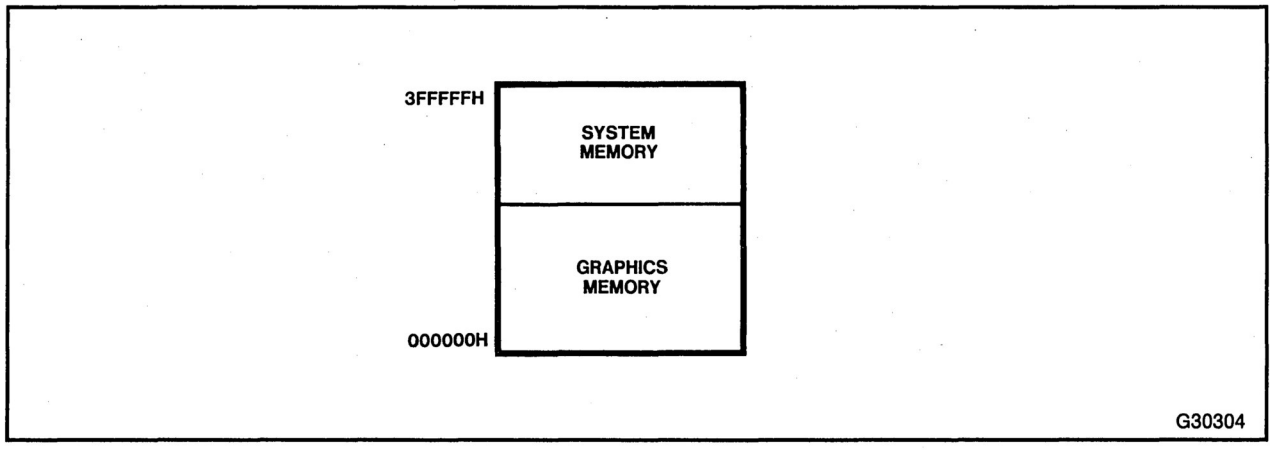


Figure 5-8. 82786 Viewpoint of Graphics Memory Map

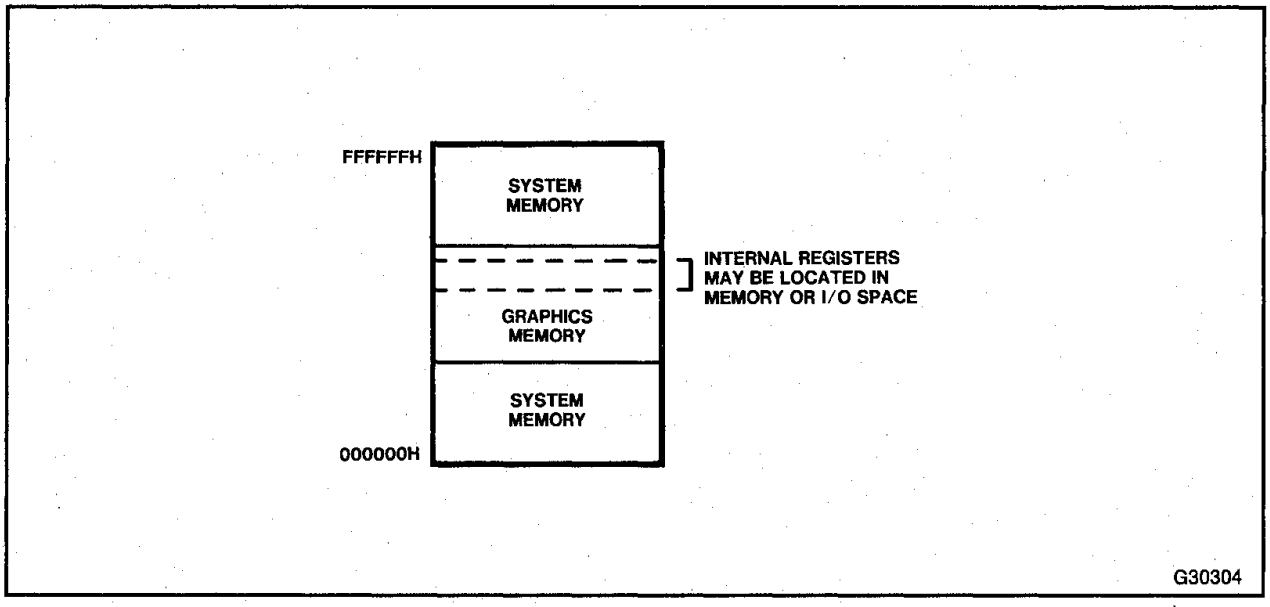


Figure 5-9. CPU Memory Map

82786 graphics memory is mapped into the 80286 address space. Also, a 3 MB portion of the 80286 memory is mapped into the 82786. Because the 80286 has two more address bits than the 82786, a tristate buffer supplies the top two address bits when the 82786 enters Master Mode (see Section 4.4.1.1 "Initiating Master Mode Interface").

Figure 5-10 also illustrates that the same memory corresponds to one set of memory addresses for the CPU and another set of addresses for the 82786 GP and DP. Although these addresses could match, it is not necessary, provided the controlling CPU software understands the relationship and converts addresses appropriately. For some cases, not matching the addresses is desirable. For example, most CPUs use the lowest memory addresses for special purposes, such as interrupt vectors. If the lowest CPU memory were 82786 memory rather than the faster system memory (for CPU access), these operations would be significantly slower. Although the real addresses do not match, the operating system for a CPU such as the 80286 can easily map the CPUs virtual addresses to the 82786 real addresses.

5.5.1 Mapping the Internal Registers

The 82786 Internal Registers can be either memory or I/O mapped. If they are memory mapped over graphics memory as shown in Figure 5-9, the CPU cannot access the 128-byte block of memory that they cover, although the GP and DP can access it. If the Internal Registers are mapped above graphics memory, over nonconfigured memory, the CPU can access all graphics memory, but the Internal Registers must be in the CPU address space that the address decoder allocates for the 82786. If the 82786 are I/O mapped, so they do not overlap any memory, the Chip Select logic for the 82786 will be larger as shown in Figure 5-11. However, memory mapping the Internal Registers allows the software slightly more flexibility in accessing the registers.

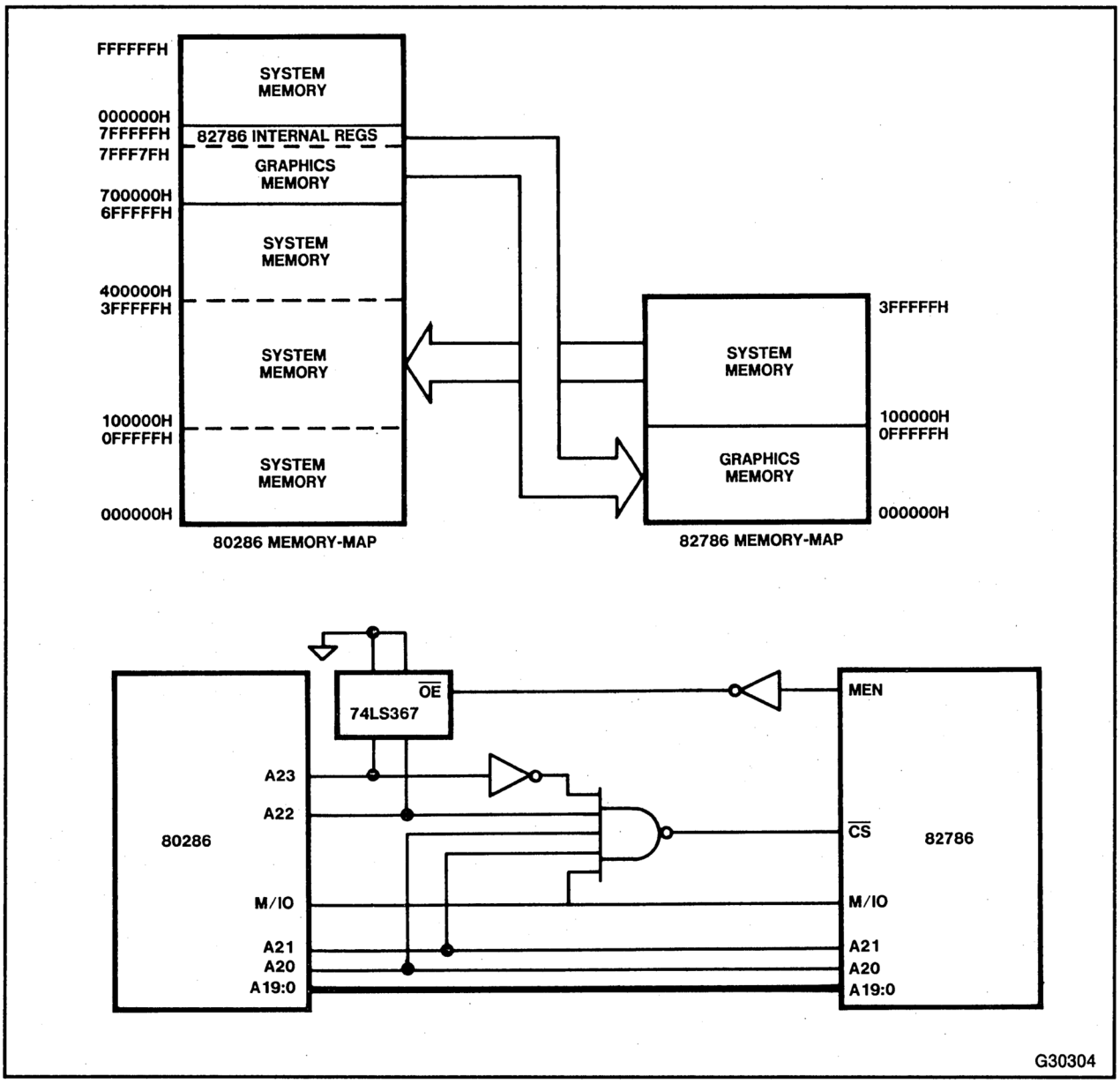


Figure 5-10. Possible Memory Map for 80286/82786 System with Memory Mapped Internal Registers

5.5.2 Alternatives for Graphics Memory Mapping

For some CPUs, the size of graphics memory can be beyond its addressing capacity or the CPU may not need to access all of graphics memory. For example, if an 82786 with 2 MBs of graphics memory is used with an 80186 processor, which can access only 1 MB of memory, the 80186 cannot access all of the configured graphics memory. However, the 80186 can access a portion of it as shown by the memory map and connection diagram in Figure 5-12. Because the 82786 has two more address bits than the 80186, a tristate buffer supplies the two highest address bits when the 82786 is in Slave Mode (see Section 4.4.2 "Slave Interface").

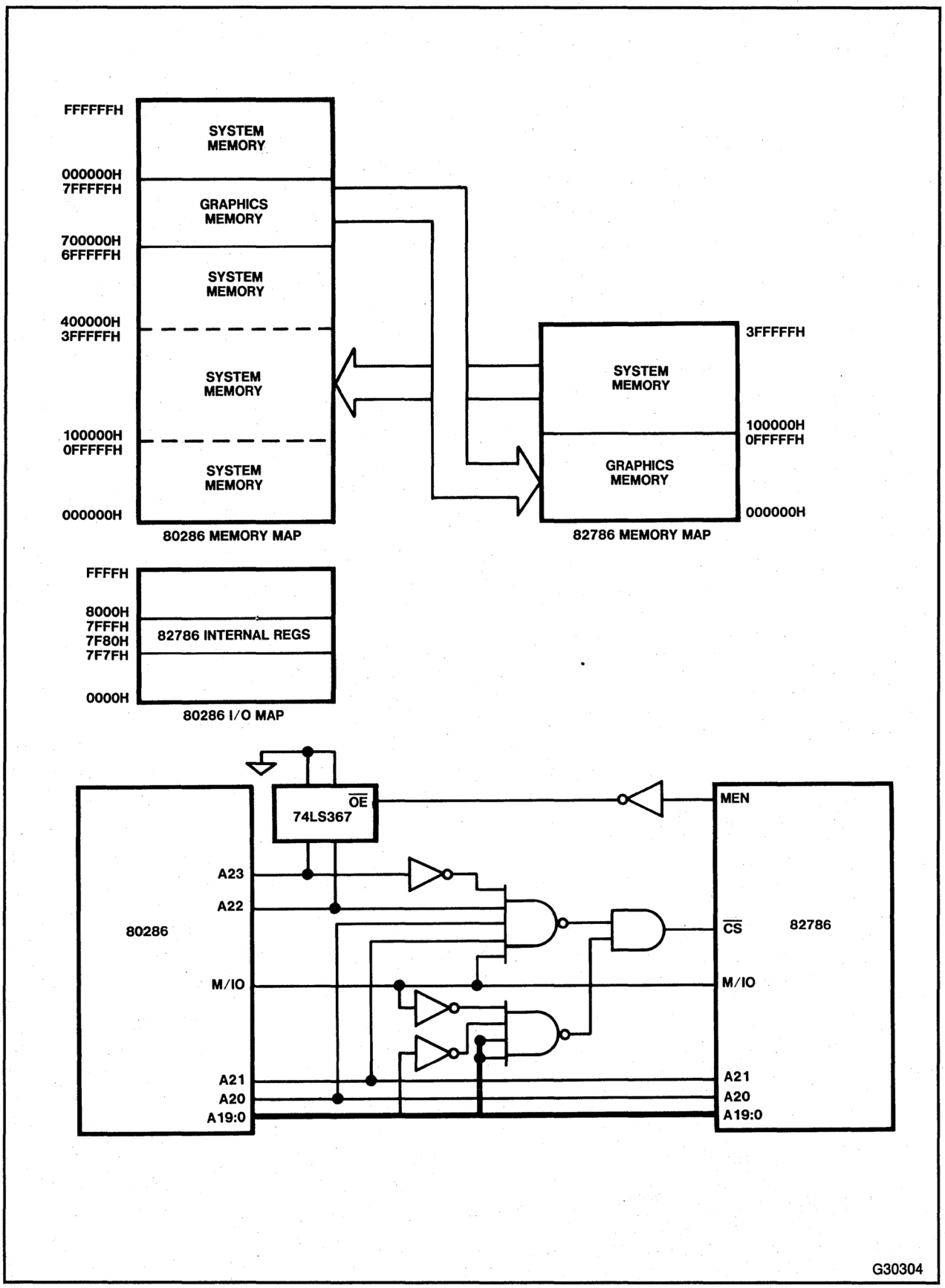


Figure 5-11. Memory Mapping for 80286/82786 System with I/O Mapped Internal Registers

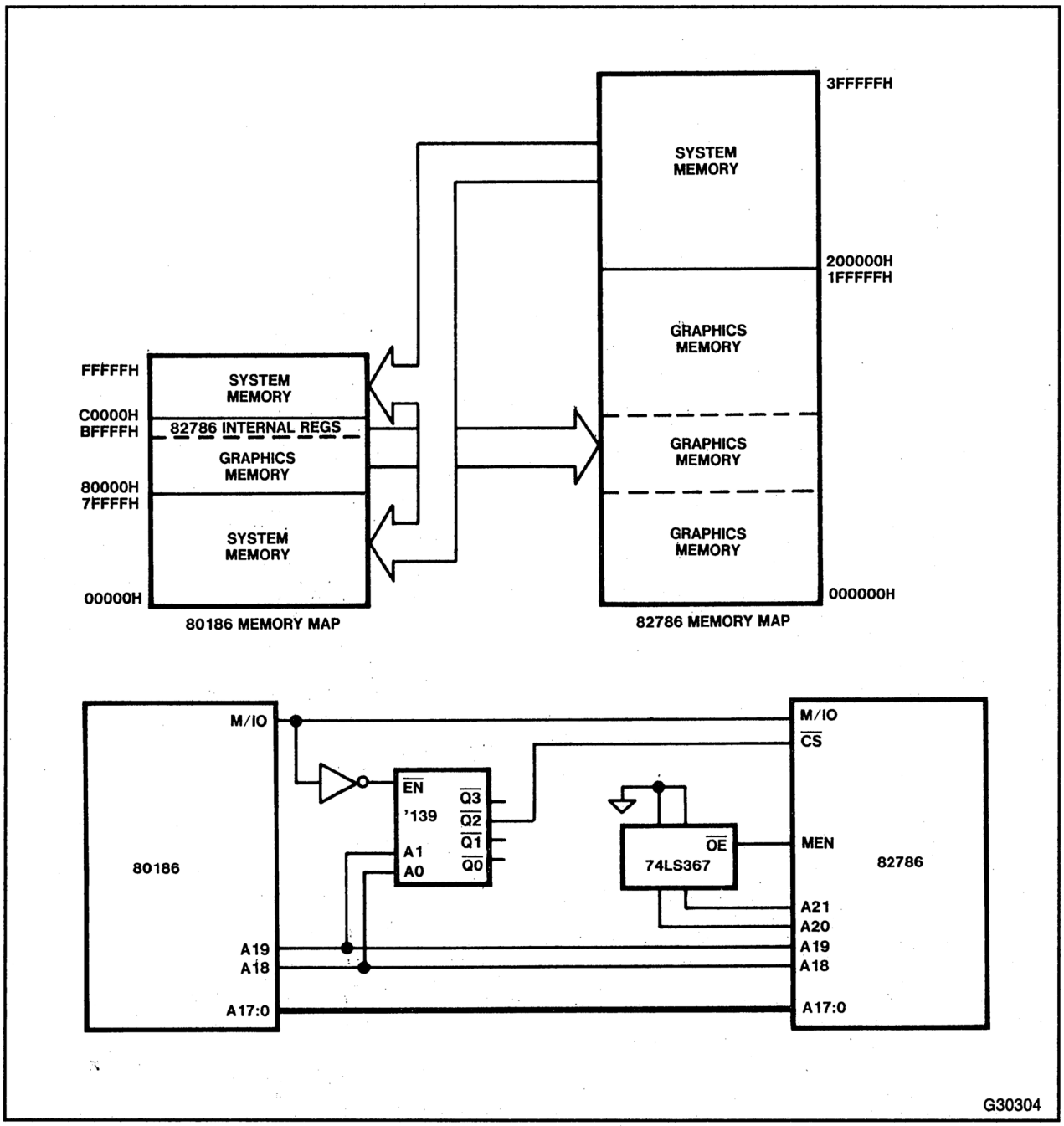


Figure 5-12. 80186 System Accesses Only a Portion of Graphics Memory

However, if the CPU must access a portion of memory, which is not directly mapped to the CPU, the 82786 Graphics Processor can be instructed (using the BitBlt command) to move portions of the graphics memory to and from the area accessible by the CPU.

An alternative mechanism is to bank switch the graphics memory areas to allow the CPU access to any area of graphics memory. Figure 5-13 displays a memory map that uses an I/O port (74LS173) to which the CPU writes the highest three bits for the 82786 slave accesses.

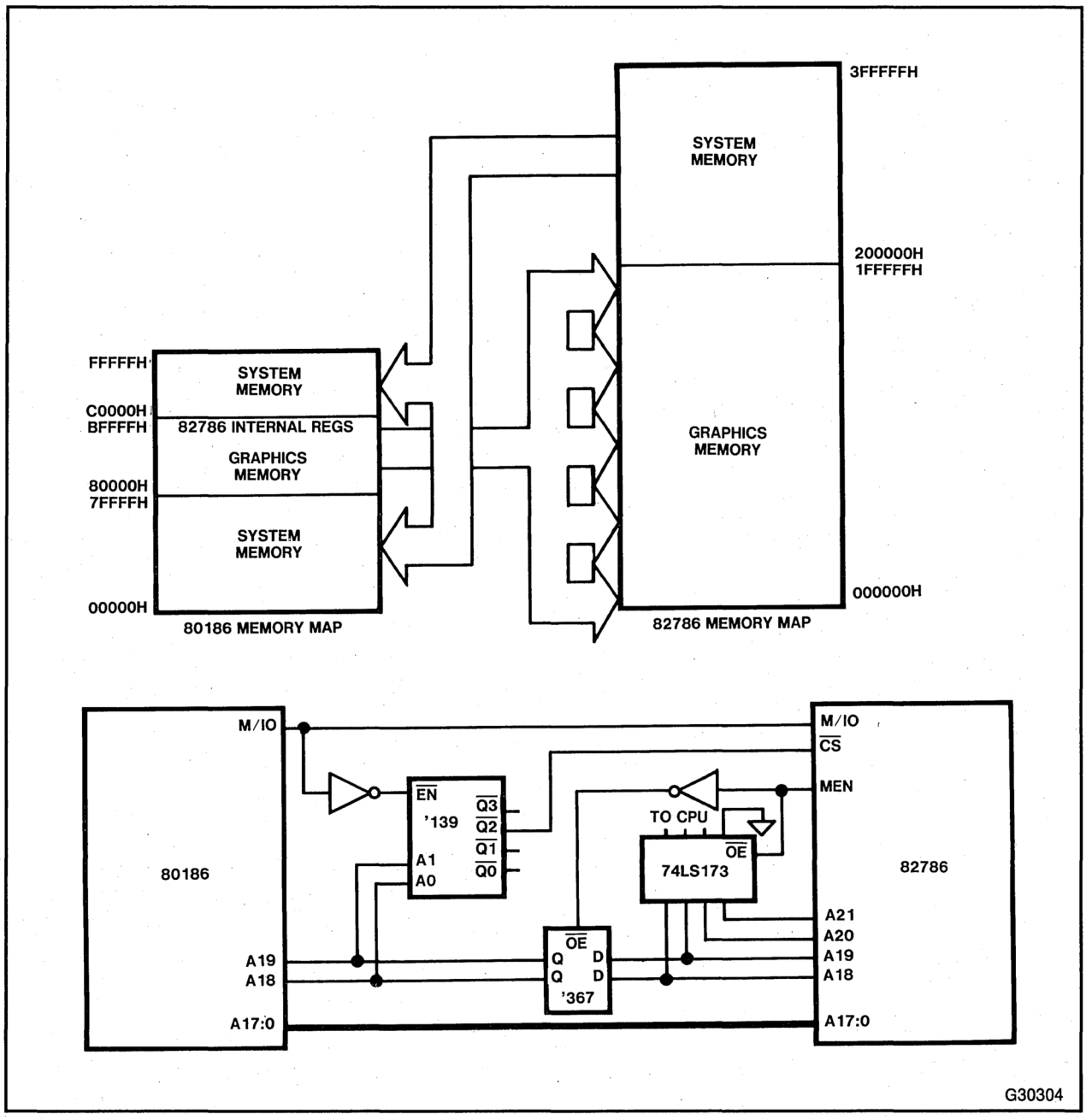


Figure 5-13. Bank Switched Memory Allows 80286 to Access all of Graphics Memory

In the preceding Figures 5-12 and 5-13, the 82786 in Master Mode can access CPU memory addresses that correspond to 82786 slave addresses. In this case, the circuit generates an 82786 Chip-Select, but the 82786 does not respond while it remains in Master Mode. As long as the READY logic goes high (it may not because the 82786 will not perform the slave access), the 82786 will complete the Master Mode cycle. By the time the 82786 returns to Slave Mode, the Chip Select will have gone away.

Video Interface 6

CHAPTER 6
VIDEO INTERFACE

The video interface connects the 82786 to the video display. The 82786 is optimized to drive CRT monitors but it also can drive other displays including LCD, plasma, and intelligent printers. Due to the wide range of displays supported and diverse requirements of each display type, this chapter restricts its detailed discussions to parameters for CRT video interfaces. CRTs provide an inexpensive method of generating moderate and high resolution monochrome and color displays. Sections 6.9 and 6.10 briefly discuss considerations for other video display interfaces. The *Intel 82786 Hardware Configuration Application Note* also discusses other types of displays.

The video interface for a CRT varies based on the CRT requirements, the resolution, and depth (bits per pixel (bpp)) of the image desired. The 82786 can be programmed to generate all CRT signals for up to 8-bpp (256 colors) displays at video rates up to 25 MHz.

Eight parallel Video Data (VDATA) Output lines provide video output from the 82786. The VDATA output can be used as eight bpp at 25 MHz on the CRT, or it can be shifted externally to boost maximum display resolution (see Section 6.4.1 "External Logic requirements." An independent Video Clock (VClk), which can be up to 25 MHz. Horizontal Synchronization (HSync) signals are programmable from 1 to 4096 cycles of the VClk. Vertical Synchronization (VSync) signals can be from 1 to 4096 scan lines. With external hardware, a color lookup table (also called a Video Palette RAM, see Section 6.2) can be used, higher display resolution can be achieved by trading off bpp for dot rate, or VRAMs can be used.

Tables 6-1 and 6-2 outline possible display configurations. The calculations assume 60 Hz refresh rate. High resolution CRTs often run slower, which enables the 82786 to generate significantly higher resolutions than these tables depict. All cases assume a CRT horizontal retrace time of 7 μS, except the 512 × 512 × 8 (10 μS) and the 640 × 400 × 8 (13 μS).

Table 6-1. Possible CRT Displays with Standard DRAMs

Bpp	Colors	Noninterlaced	Interlaced
8	256	512 × 512 640 × 400 640 × 480	900 × 675 900 × 675 900 × 675
4	16	870 × 650	1290 × 968
2	4	1144 × 860	1740 × 1302
1	monochrome	1472 × 1104	2288 × 1716

Table 6-2. Possible CRT Displays with VRAMs*

Bpp	Colors	Noninterlaced
8	256	1024 × 1024
4	16	2048 × 1024
2	4	2048 × 2048
1	monochrome	4096 × 2048

* For 64K × 4; with 256K × 4 higher resolutions are supported.

Systems with multiple 82786s can generate even higher resolutions with more colors. For example, two 82786s can create a noninterlaced, 1144 × 860, 16-color display using DRAMs. For details, refer to Section 6.5 "Greater Resolution With Multiple 82786s."

6.1 CRT INTERFACES

CRT monitors can use a variety of interfaces. Some use TTL-level inputs, others require analog inputs. Some use separate color inputs (red, green, and blue called RGB) and separate horizontal synchronization (HSync) and vertical synchronization (VSync) while others require that some or all of these signals be combined into composite signals. This chapter discusses an RGB monitor with separate HSync and VSync signals. Standard techniques can be used to convert these separate signals into composite signals to meet the requirements of other displays.

The Video Clock (VClk) required by the 82786 can be generated by a simple oscillator with TTL outputs. Alternatively, the VClk can be tied to the bus clock (CLK) or any other available clock if they are to run at the same speed.

6.1.1 CRTs with TTL-Level Inputs

The simplest interface connects CRTs with TTL-level inputs. The 82786 can generate these signals directly as in Figure 6-1.

However, the drive requirements of the CRT and cabling may make it necessary to buffer the signals as shown in Figure 6-2.

Both monitors in these examples use four bits per pixel (bpp) of color information, which allows 16 available colors. Both CRTs can use the 1-, 2-, and 4-bpp Modes, but cannot use the 8-bpp Mode to yield 256 colors. A monochrome monitor with only one TTL-level input can be connected directly to VDATA0 and use the 1-bpp Mode, but then it cannot use any of the higher bpp Modes.

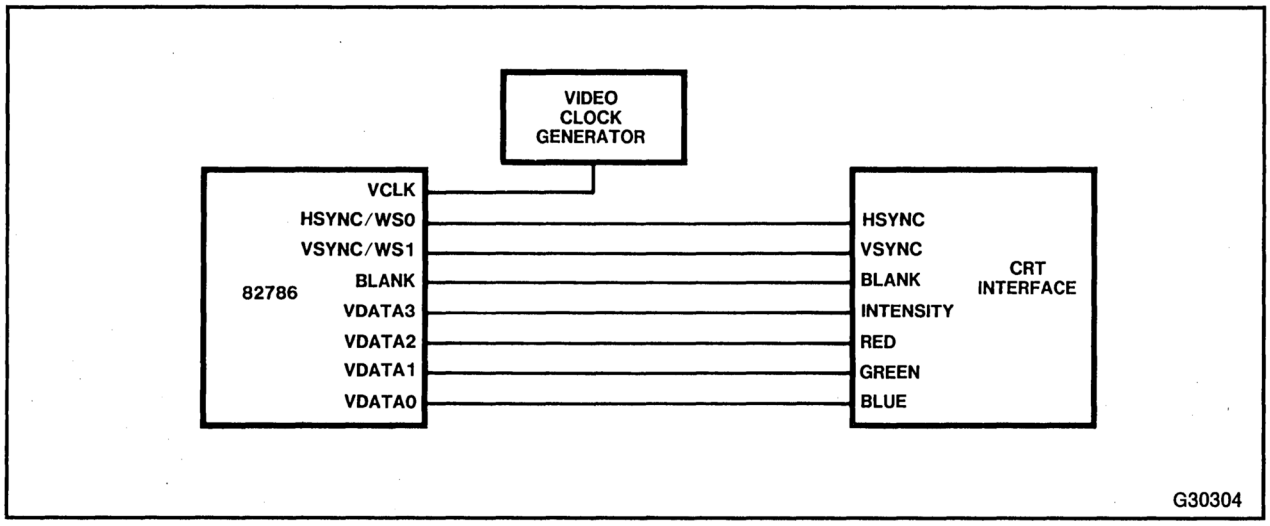


Figure 6-1. 82786 Drives CRT with TTL-Level Inputs

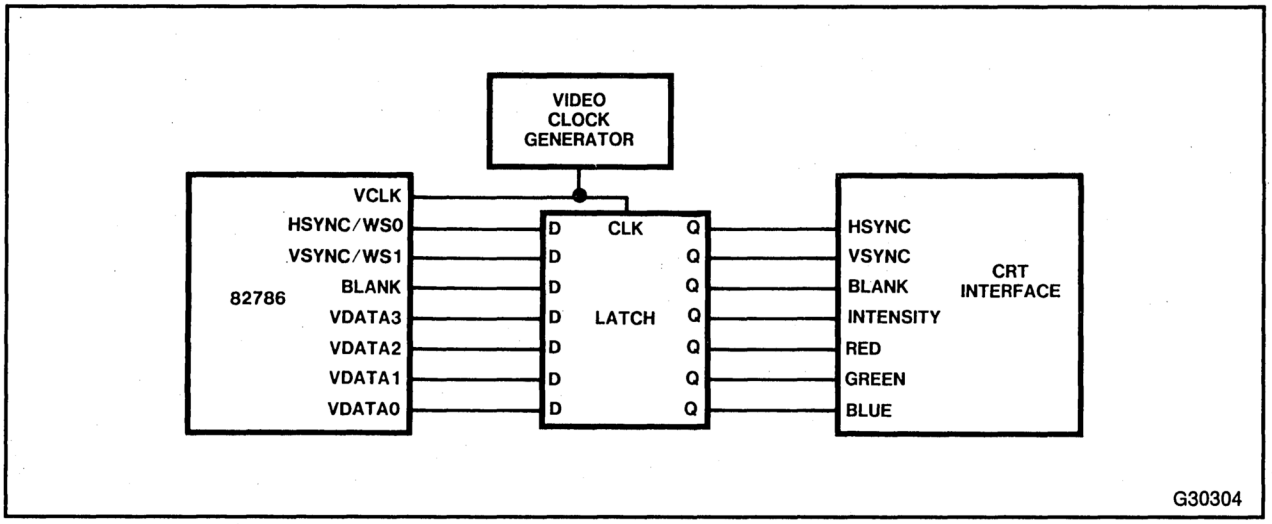


Figure 6-2. Buffer Used to Drive TTL-Input CRT Interface

6.1.2 CRTs with Analog Inputs

The 8-bits per pixel (bpp) Mode of the 82786 usually requires using a CRT with analog inputs. Signals for color CRTs with three separate analog video inputs, (red, green, and blue) can be generated using three digital-to-analog converters as shown in Figure 6-3.

Often the digital-to-analog converters can be constructed using simple resistor ladders as shown in Figure 6-4.

With eight bpp, usually three bits select red, three select green, and two select blue. More bits usually are used for red and green because human eyes are much more sensitive to variations of red and green than of blue. These configurations can take advantage of all the 82786 modes: 1, 2, 4, and 8 bpp. The VDATA pins can be assigned to the three colors in any manner desired. Figure 6-5 illustrates an assignment that ensures a variety of colors are available for each mode.

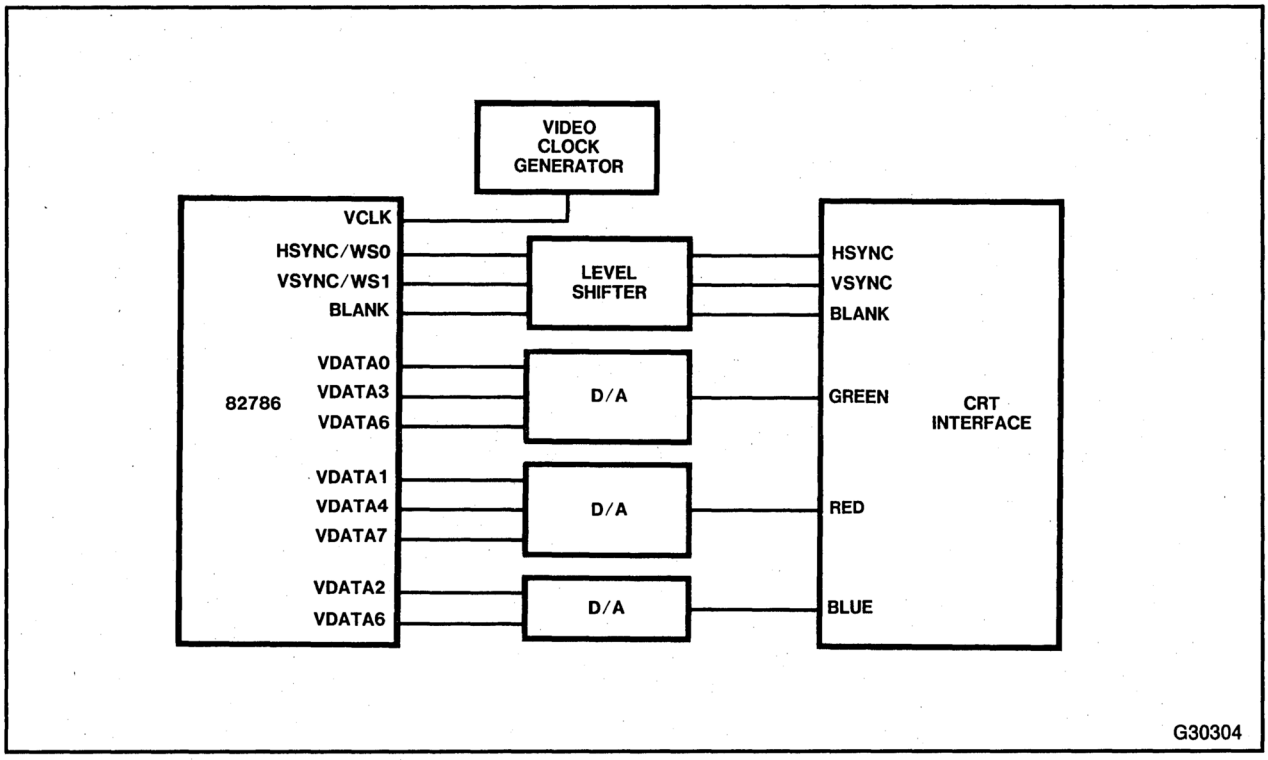


Figure 6-3. Analog CRT Interface Allows 256 Colors

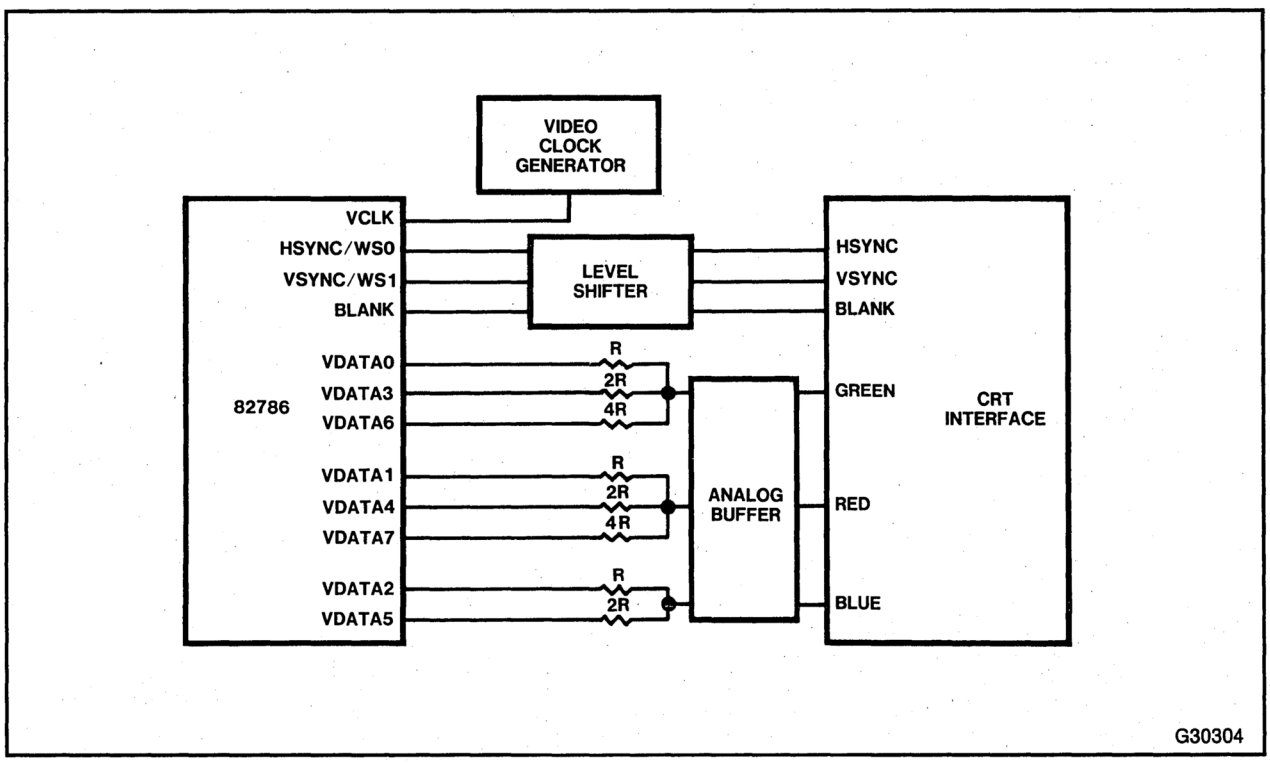


Figure 6-4. Resistor Ladder

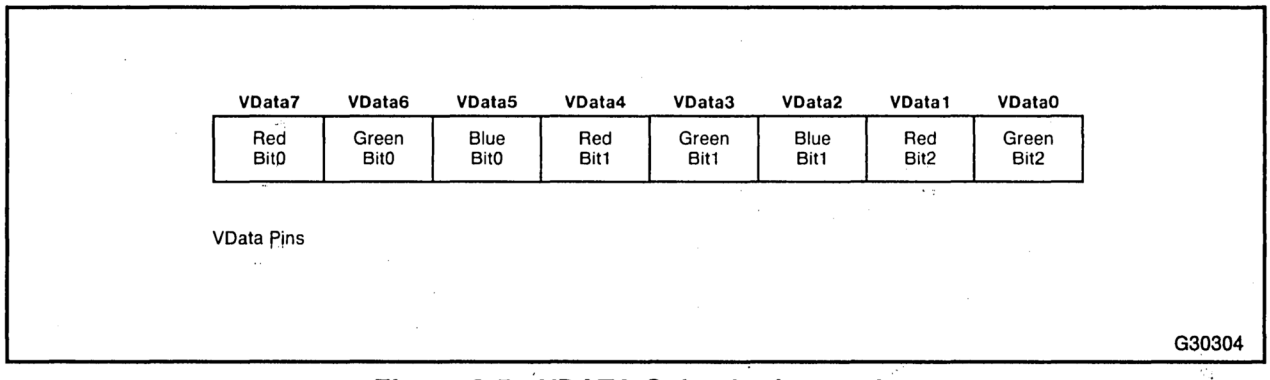


Figure 6-5. VDATA Color Assignments

In Figure 6-5, a non-Accelerated Mode example shows the most significant green bit connected to VDATA0. In the 1-bpp Mode, the most significant green bit is controlled while the other bits are set to a constant level by the Display Processor Pad Register (see Section 3.3.3 "Pad Registers"). For example, if the pad bits are all set to zero, then a green and black image is shown in 1-bpp windows.

With 2-bpp Mode, the most significant green and red bits are controlled while the remaining bits are padded to a constant value. For example, if the pad bits are set to zero, the colors black, green, red, and yellow are available with two bpp.

The 4-bpp windows contain two green bits and the most significant red and blue bits to provide 16 available colors.

The 8-bpp windows allow control of all eight bits to provide 256 available colors.

Figure 3-14 illustrates the arrangement of pixels on the VDATA pins during Accelerated Video Modes.

6.2 USING A COLOR LOOKUP TABLE/VIDEO PALETTE RAM

Color lookup tables, also called Video Palette RAMs, allow more colors to be available with minimum bits per pixel (bpp), which minimizes display memory requirements for the bitmap. For example, a system using 16 bits of color information allows 65,536 different colors. However, rarely will such a system display all 65,536 colors on the screen simultaneously. Perhaps a maximum of 256 colors displayed simultaneously is more realistic, if the 256 selections can be any combination of the 65,536 available colors. Color lookup tables can create such an efficient and effective system as outlined in Figure 6-6.

The color lookup table can be loaded with up to 256 16-bit colors, which can be controlled by an 8-bpp bitmap. The lookup table is loaded by first writing the location into the 82786 Default Video Register. Then a 4-bit color value is loaded into the latch along with color-select information. In one load operation, this 4-bit color value can be placed into any combination of the red, green, and/or blue tables.

The host CPU loads the 16-bit colors into the lookup table. To load a color into a specific location in the lookup table, the 82786 Display Processor (DP) can output the 8-bit address

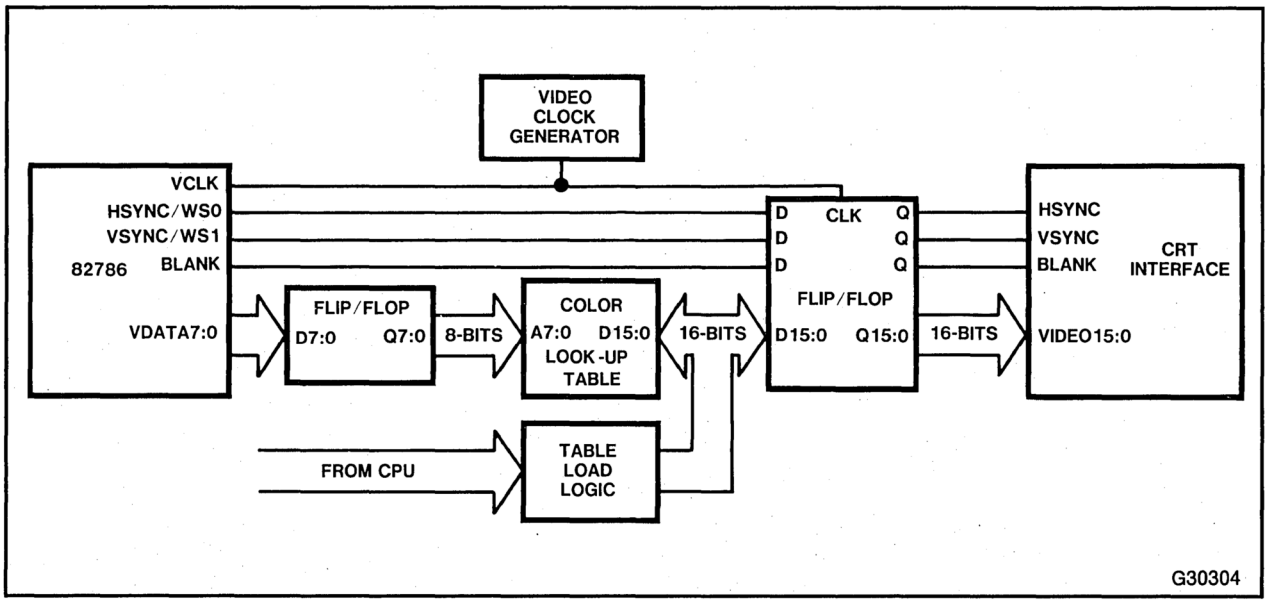


Figure 6-6. Color Lookup Table

on the 8 VDATA pins during the horizontal and vertical Blank times by programming the Default Video Register (see Section 3.3.1 "Display Processor Internal Registers"). Then, the CPU can load the color value into the 16-bit latch. The circuitry in Figure 6-7 automatically writes the 16-bit value into the lookup table during the next HSync time.

The CPU should generate the latch enable input so the latch can be mapped into memory or I/O space and loaded by a CPU write. The register between the 82786 and the Video Palette RAM allows the use of a RAM with a slower access time. This register is not necessary if a faster RAM is used.

The CPU program should wait until the color is loaded into the lookup table before loading the next color. To ensure the program waits, route the lookup table loading signal through a port which the CPU can poll. Sample assembly language code for this configuration is shown below.

```
Wait:   in      al,StatusPort          ; read port
        test    al,LookupLoadingBit    ; test LookupLoading bit
        jnz     Wait                   ; wait until last load completed
        mov     ax,EightBitAddr        ; get 8-bit value to load
        mov     DefaultVDATA,ax        ; make 82786 output during Blank
        mov     ax,SixteenBitColor     ; get 16-bit color
        out     LookupLatch,ax         ; write color into latch
```

Another way to ensure that the program waits is for the CPU program to delay a sufficient amount of time to ensure that HSync occurs before it writes the next value.

Hybrid circuits like the one shown in Figure 6-8 can combine the functions of the lookup table, analog-to-digital conversions, and voltage shift for composite synchronization signals into one package.

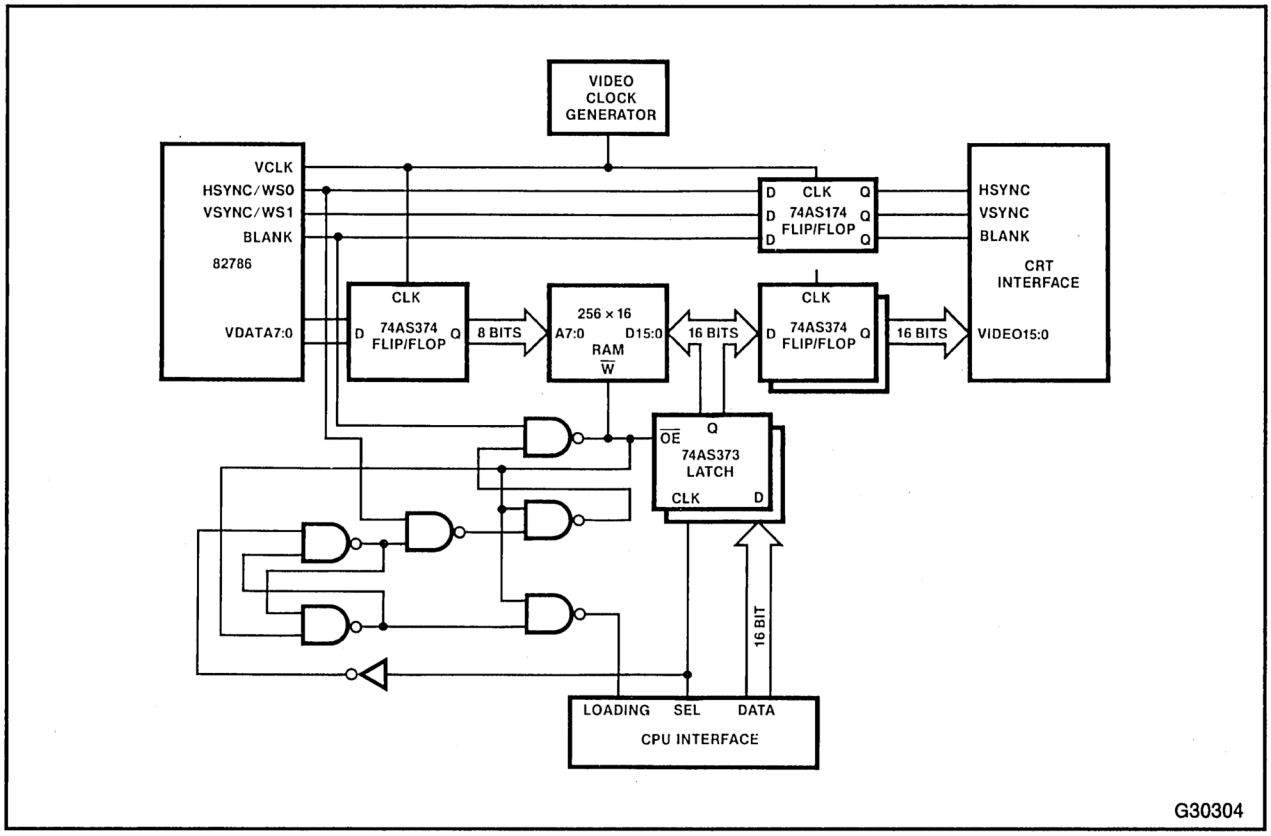


Figure 6-7. Color Lookup Table Circuit Generates 16 Video Bits from 8

6.3 WINDOW STATUS SIGNALS

A graphics system design may require that the video data bits for different windows be interpreted in different ways. For example, the attributes controlled by various video data bits may need to be changed between windows for different tasks or the number of bits per pixel (bpp). To accomplish this task, two Window Status (WSt) bits are available externally. The value of these WSt bits can be individually programmed for each window. The Window Status Pins always contain the value for the window that the display is currently scanning. The software must place the 2-bit value for each window in each Tile Descriptor of the Strip Descriptor (see Table 3-1 in Section 3.1.3.2 "Strip Descriptor Format").

6.3.1 Controlling the Cursor

The Cursor Status (CSt) bits in the CsrMode Register in the Display Processor Control Register Block can be programmed with a value for the Window Status (WSt) bits to override the WSt bits for the portion of the display where the cursor resides. For details on the CSt bits, refer to Table 3-8 in Section 3.3.2 "Display Control Block Registers."

The WSt bits are multiplexed onto the HSync and VSync pins because they are applicable only during the visible display time, and HSync and VSync are applicable only during the nonvisible display time. Blank can be used to demultiplex these pins as shown in Figure 6-9.

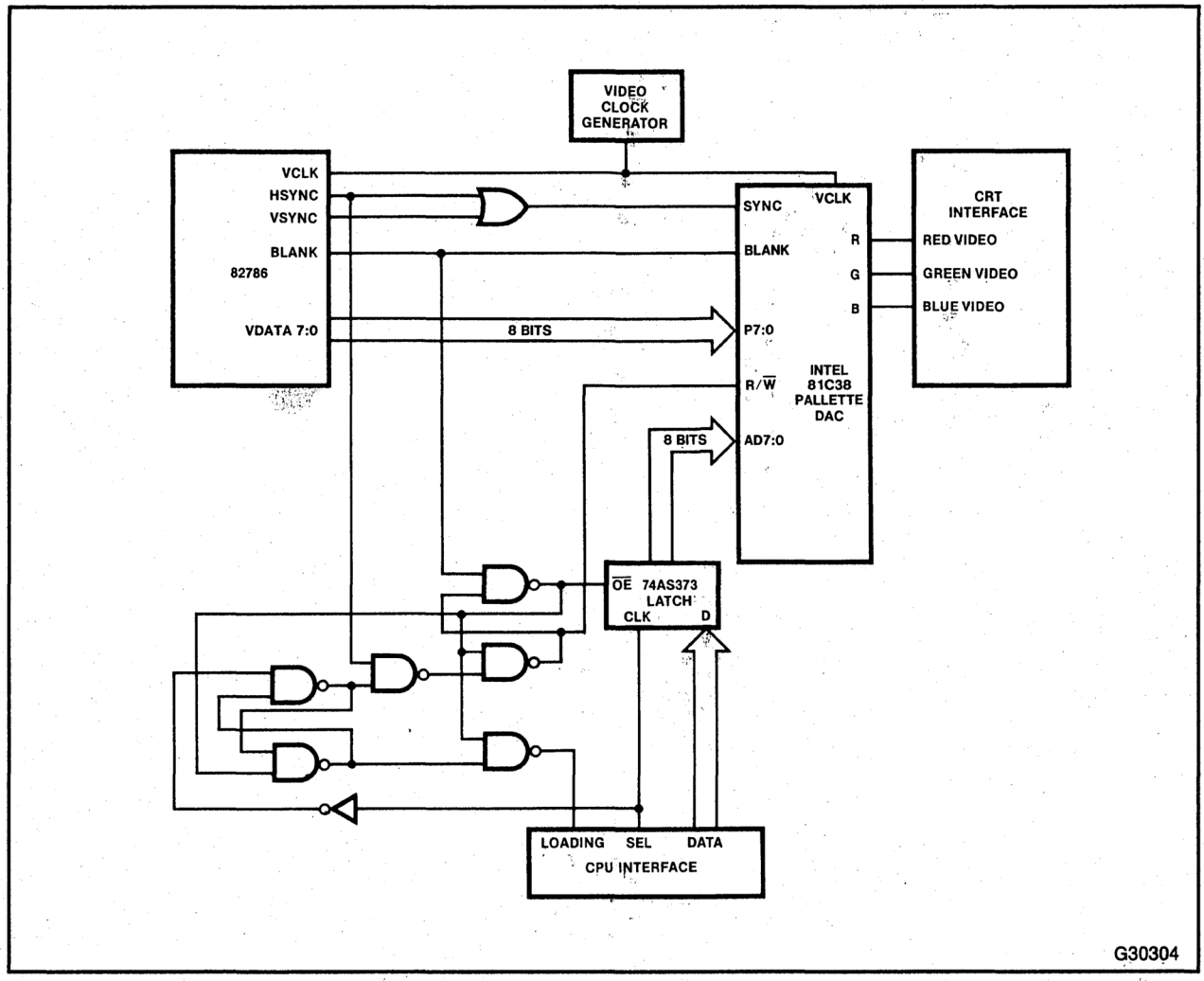


Figure 6-8. Hybrid Color Lookup Table and DAC Simplifies Interface

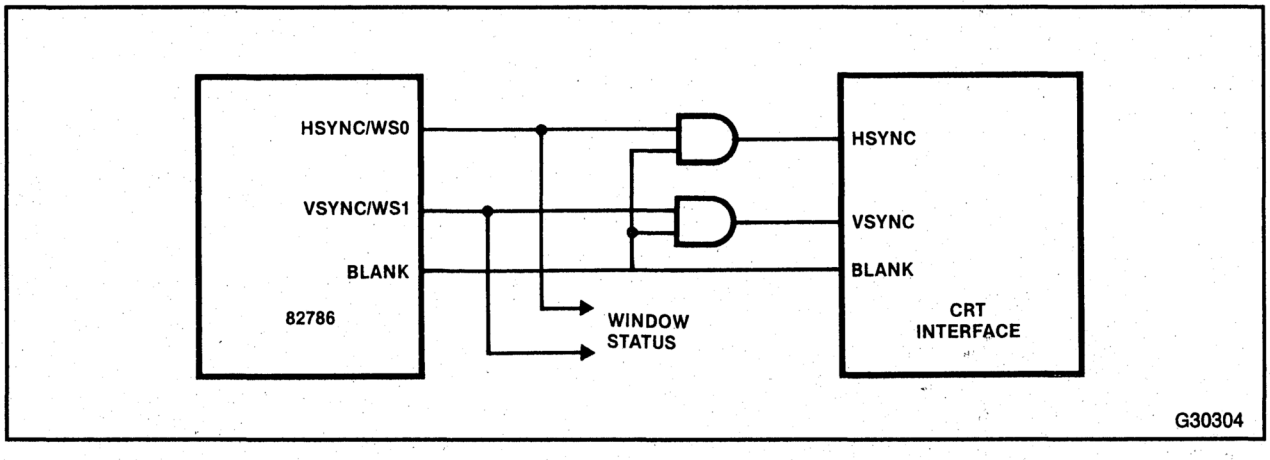


Figure 6-9. Blank Demultiplexes Window Status Pins

The Window Status Enable (W) bit in the CRTMode Register enables the WSt bits so they become multiplexed onto the HSync and VSync signals (see Table 3-8 in Section 3.3.2 Display Control Block Registers"). This bit must be set when the Window Status signals are used. In systems where the WSt bits are not needed, this bit can be reset so that the HSync and VSync pins remain low during the visible display. This allows simple systems to use HSync and VSync directly, which eliminates the need to AND these pins with Blank.

6.3.2 Window Status Bits Segment Lookup Table for Multiple Windows

Normally, interpreting the Video Data bits (VDATA) differently for each of four different types of 8-bpp windows requires four different lookup tables. Alternatively, one large lookup table (1024 words) can be divided into four areas, one for each window interpretation, with the Window Status (WSt) bits selecting the area of the lookup table for each window interpretation. Essentially, four lookup tables exists, one for each type of windows as shown in Figure 6-10.

The system in Figure 6-10 also requires circuitry similar to that in Figure 6-6 to load the lookup table. The CPU must generate the Window Status inputs for the lookup table directly when the RAM is loaded because these pins can not be programmed during the Blank time like the VDATA pins.

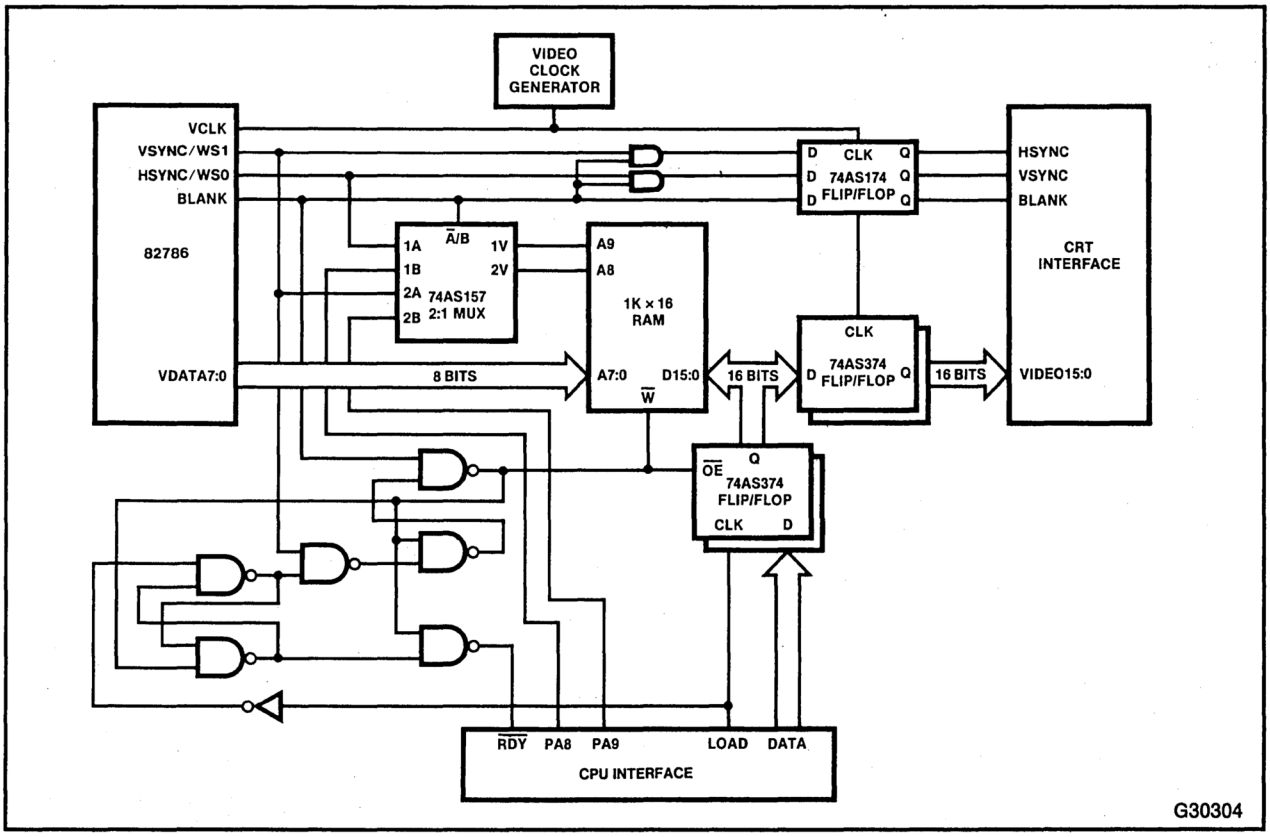


Figure 6-10. Four Color Lookup Tables Selectable by Window Status Outputs

The WSt bits also allow 1-, 2-, 4-, and 8-bpp windows to use different lookup tables. A 1024 word lookup table can be split into four areas as in Figure 6-8. Two of the areas can be used for two separate 8-bpp lookup tables, with the remaining two shared by the 1-, 2-, and 4-bpp windows for two separate lookup tables for each of 1, 2, and 4 bpp. The pad bits can subdivide this second area into separate tables for 1, 2, and 4 bpp windows. Finally, this same table could also be used for twelve lookup tables, four each for 1-, 2-, and 4-bpp windows.

6.4 HIGH RESOLUTIONS WITH STANDARD DRAMs

The Video Clock (VClk) rate on the 82786 can be a maximum of 25 MHz. For a noninterlaced display, refreshed 60 times per second, this limits the resolution to 512 × 512, 640 × 480, or equivalent displays. Some graphics system designs may require more detailed displays requiring more resolution. The most cost-effective solution trades off the number of bits per pixel (bpp) for higher resolution. This solution is especially effective for monochrome displays when higher resolution requirements override the need for 256 gray shades.

6.4.1 External Logic Requirements

With external logic, the 82786 supports Accelerated Video Modes in which video data rates exceed 25 MHz. Figure 6-11 illustrates how a video data rate of up to 50 MHz can be obtained with four bpp (16 colors).

In Figure 6-11, the 82786 outputs eight bits of video data at 25 MHz. The external multiplexer switches between the low four bits and the high four bits at a rate of 50 MHz. The register before the multiplexer ensures that enough setup time exists for the multiplexer. The register after the multiplexer ensures that the video data out has smooth transitions.

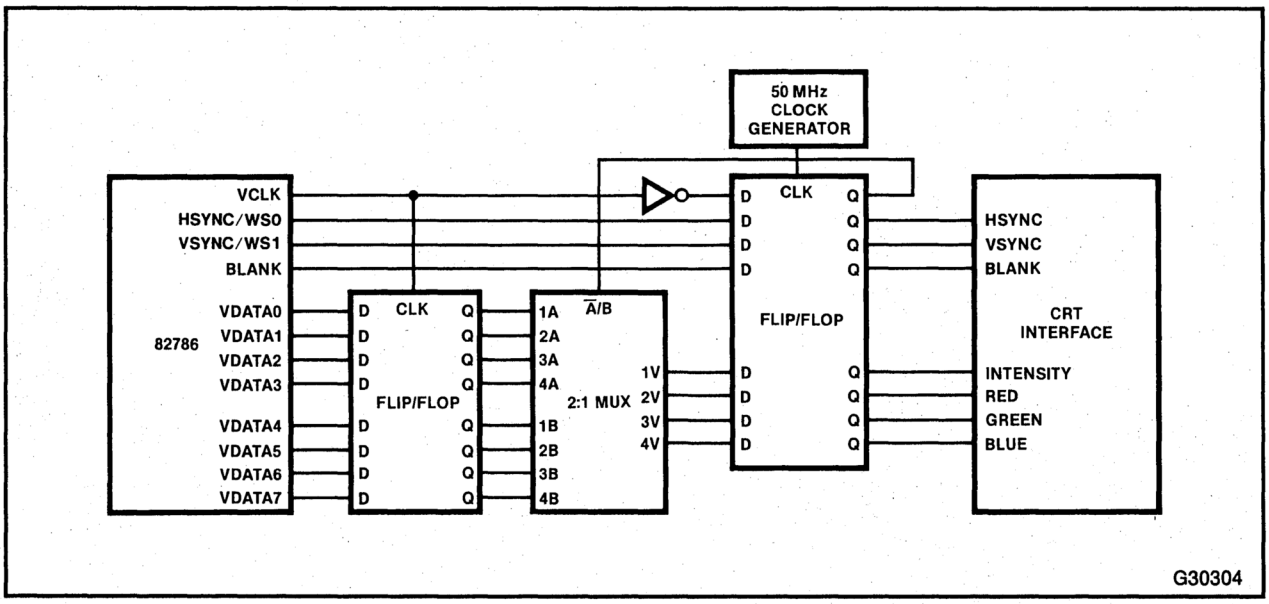


Figure 6-11. External Multiplexer Allows up to 50 MHz Video with 4 Bpp

The circuit uses an inverter and one register stage to divide the 50 MHz clock by 2 to create the 25 MHz video clock for the 82786. Instead of the separate multiplexer, a chip can be used which contains the multiplexer and the register in the same package.

Figure 3-14 illustrates the arrangement of the pixels on the VDATA pins when using Accelerated Modes.

6.4.2 Software Changes Required for High Resolution

High Resolution achieved by increased video rate requires minimal software changes. The Graphics Processor (GP) is programmed identically and manipulates the bitmaps in the conventional manner (although it does not make sense to use 8-bpp bitmaps because they cannot be displayed). The Display Processor (DP) programming differs slightly. The Accelerated Video Control (AA) bits in the CRTMode Register for High-Speed video are set to 01 (see Table 6-5 in Section 6.10.3 "Video Interface Parameters"). The Video Timing Registers HSyncStp, HFldStrt, HfldStp, and LineLen are programmed for half the number of dot clocks because the 82786 VClk is half the speed of the pixel dot clock in High-Speed Accelerated Video Mode.

Tile Descriptors for each Strip Descriptor also change slightly. Windows are programmed for the same number of bits per pixel (bpp) and Fetch Count as they would be for Nonaccelerated Modes. However, horizontal resolution for window positioning is reduced to even pixel boundaries of 2, 4 or 8 bits based on the current Accelerated Mode. Both even and odd pixels are output on the VDATA pins simultaneously. Mixing windows is not possible during a single VClk. Table 6-3 lists valid values for the StartBits and StopBits. The Accelerated Modes do not permit all possible bitmap depths because fewer than 8 bpp are available to the display. For details on Tile Descriptors, refer to Section 3.1.3.2 "Strip Descriptor Format."

Table 6-3. Valid StartBit and StopBit Values

Bitmap Depth Bpp	Acceleration							
	None (25 MHz)		High-Speed (50 MHz)		Very High-Speed (100 MHz)		Super High-Speed (200 MHz)	
	StartBits	StopBits	StartBits	StopBits	StartBits	StopBits	StartBits	StopBits
1	0–15	0–15	odd	even	15,11,7,3	12,8,4,0	15,7	8,0
2	odd	even	15,11,7,3	12,8,4,0	15,7	8,0	—	—
4	15,11,7,3	12,8,4,0	15,7	8,0	—	—	—	—
8	15,7	8,0	—	—	—	—	—	—

Vertically, the windows can be positioned at any pixel. The 1-pixel horizontal and vertical borders do not change positioning resolution in Accelerated Modes. High-Speed Accelerated Video Mode also requires that the background field tiles, defined by the F bit in the Tile Descriptor, are programmed with half the number of actual pixels for the pixel count register (BPP, StartBits, and StopBits; see Table 3-1 in Section 3.1.3.2 "Strip Descriptor Format"), which again restricts horizontal positioning to a 2-pixel resolution in High-Speed Mode.

6.4.3 Cursor Control

Horizontal cursor positioning also is affected in Accelerated Video Modes. In High-Speed Mode, the horizontal positioning must be programmed as half the actual value, which restricts the horizontal resolution to two pixels. Vertically, the cursor is programmed as normal. With the block cursor being a 1-bit per pixel (bpp) region, every other horizontal pixel reflects the cursor pad value. Although simple cursor patterns are possible, arbitrary shapes are not possible with the block cursor. For this reason, you may want to create the cursor in software when using a high resolution Accelerated Video Mode, rather than use the 82786 hardware cursor. The crosshair cursor works well in Accelerated Modes, although in High-Speed Mode the vertical and horizontal lines become two pixels wide and horizontal positioning also is limited to two pixels.

When using external hardware to create the cursor, you can program the cursor to be invisible (transparent and all background) and use the Cursor Window Status signals (CSt) to activate the external hardware (see CsrStyle and CsrPad Bytes of the CsrMode Register in Table 3-8 of Section 3.3.2 "Display Control Block Registers").

Use of the High-Speed Accelerated Video Mode also effects the horizontal zoom capability. Rather than replicating each individual pixel, pairs of pixels are replicated. Vertical zoom works as normal.

6.4.4 Zoom In Accelerated Modes

If zoomed windows are required in Accelerated Modes, external logic is needed to slow down the rate at which data is clocked out of the serializing latch based on the zoom factor. The 82786 holds the same value on the VDATA pins for the number of VClk cycles given by the zoom factor. The multiplexer/shifter must be slowed down under control of the Window Status pins to maintain the correct data flow for zoomed windows.

6.4.5 Examples of High Resolution Configurations

With two bits per pixel (bpp), the video rate can be accelerated to 100 MHz, as shown in Figure 6-12.

A shift register multiplexes the eight video bits from the 82786 into four 2-bit data streams. A flip/flop divides the 100 MHz clock by four. Every fourth clock the 82786 VClk is raised and the shift registers are loaded with the previous 82786 VDATA. The video data is delayed two cycles by this circuit while the HSync, VSync, and Blank are delayed one. This is not a

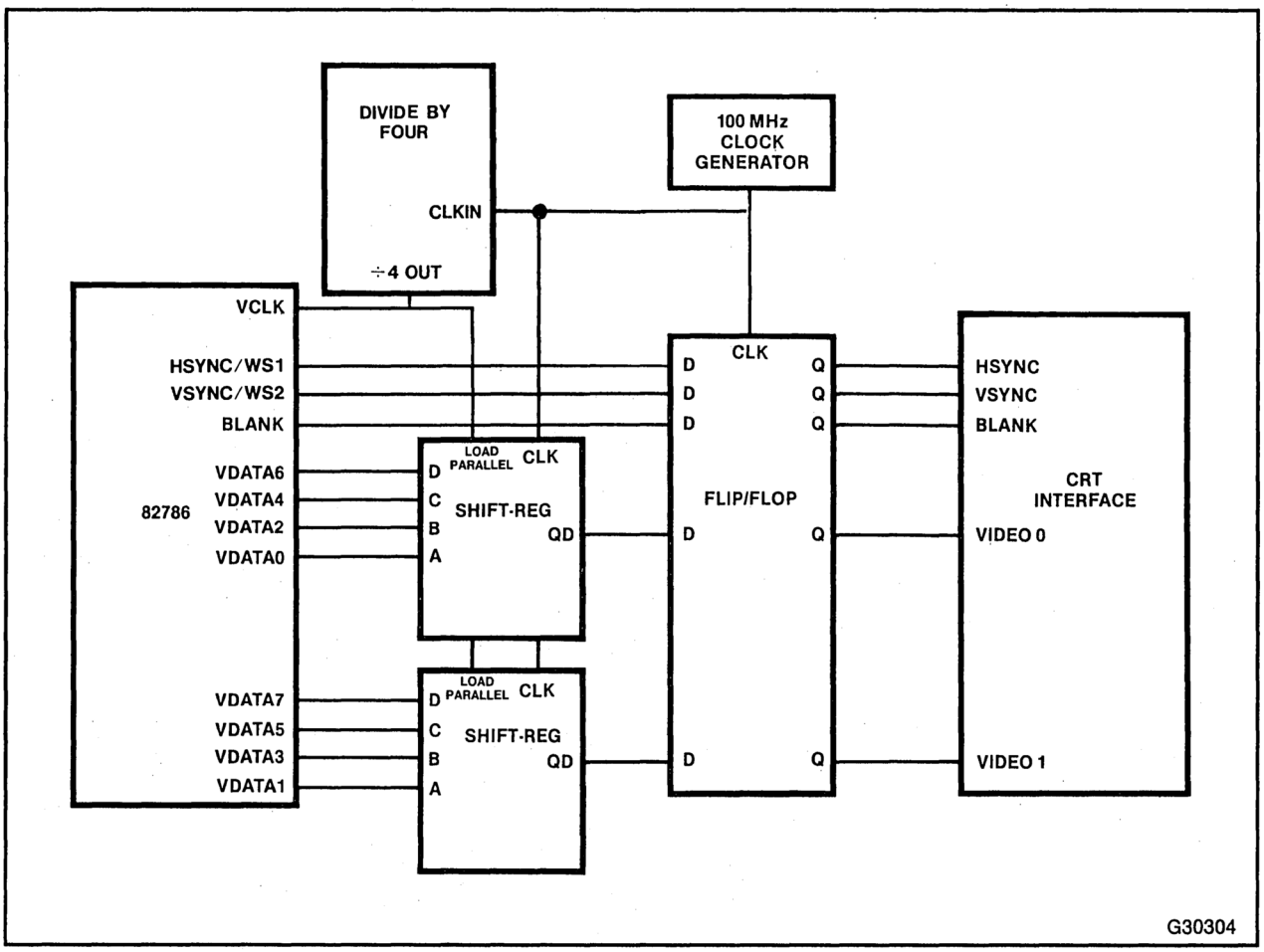


Figure 6-12. Configuration for Video Data Rates up to 100 MHz

problem if the 82786 is programmed to generate the correct HSync and VSync. The 82786 is limited to positioning the HSync and VSync transitions at multiples of four pixels. If more accurate positioning is required, extra flip/flops can be used to delay HSync and VSync for more cycles.

The timing in Figure 6-12 is very tight and the circuit may not operate at 100 MHz over all operating temperatures. The limiting speed path is the shift register parallel-load time (delay from clock to outputs valid), which must meet the setup time of the latch.

Figure 6-13 shows a configuration for video data rates of up to 200 MHz with 1 bpp. Unfortunately, no TTL-level logic is available today that runs at the speeds required for 200 MHz. ECL or some other high speed logic must generate video at these high rates. The configuration in Figure 6-13 converts the video data signals from the 82786 TTL levels to ECL levels and then uses ECL shift registers to generate the 200 MHz signal.

The software for the configurations in Figures 6-12 and 6-13 requires changes similar to those for the configuration in Figure 6-11. The StartBits and StopBits are restricted as shown in Table 6-3. The pixel count for Tile Descriptor parameters (Bpp, StartBit, and StopBit) is one-fourth for 100 MHz or one-eighth for 200 MHz the actual size (also see the F bit in Table 3-1 Tile Descriptor Parameters in Section 3.1.3.2 "Strip Descriptor Format").

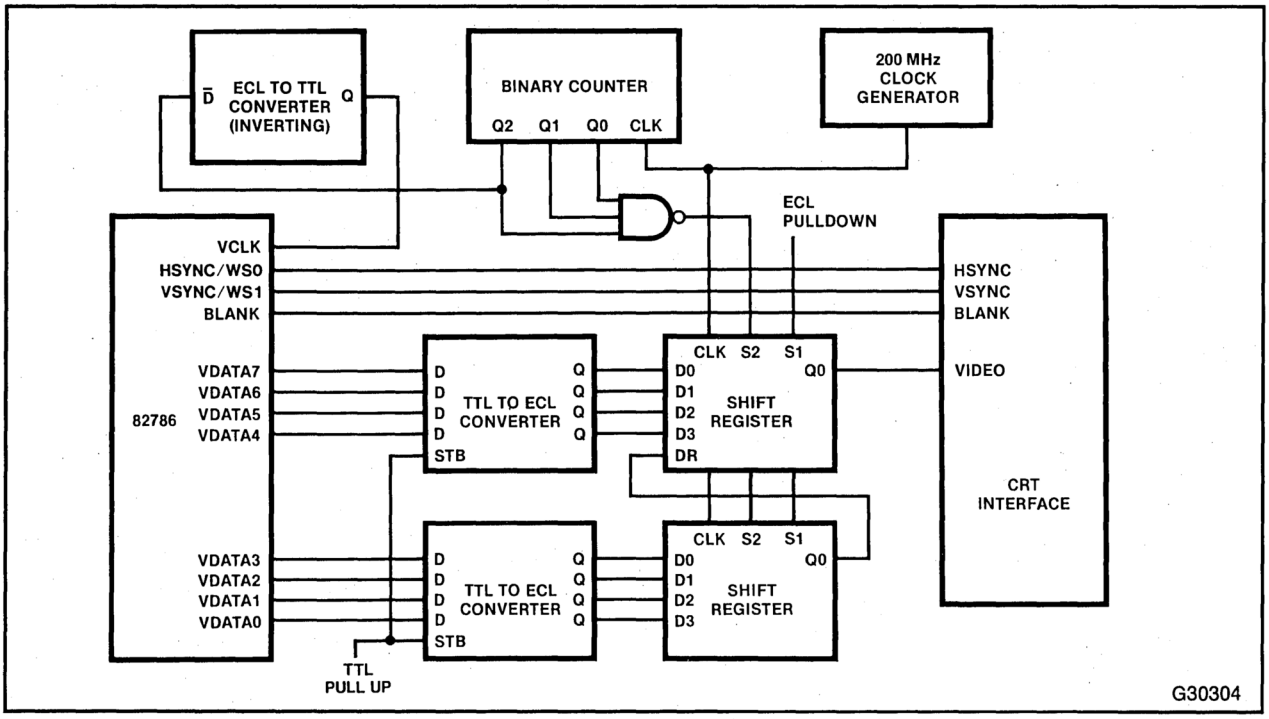


Figure 6-13. External ECL Shift Register Allows up to 200 MHz Video with 1 Bpp

Horizontal positioning also is restricted to four pixels for 100 MHz and eight pixels for 200 MHz. The Accelerated Video Control Mode bits (AA) are programmed with a value of 10 at 100 MHz and a value of 11 for 200 MHz. Figure 3-14 illustrates the arrangement of pixels on the VDATA pins when using Accelerated Modes.

After the video signals are accelerated to these higher speeds, color lookup tables and analog-to-digital converters may be used. The circuits in the previous sections must be adapted for these higher speeds.

6.5 GREATER RESOLUTION WITH MULTIPLE 82786s

Combining multiple 82786s in a system can provide greater resolution or colors than a system with one 82786. Figure 6-14 shows two 82786s used to generate 16 bpp at a 25 MHz video rate. This configuration would allow a 640 × 480 display with 65,536 colors.

In the system in Figure 6-14, both video signals must be kept synchronized. To do this, one 82786 generates master video HSync and VSync while the other 82786 is the slave receiving HSync and VSync signals from the master. To program the master 82786, set the HSync/VSync (S) and Blank (B) Slave Mode bits in the CRTMode Register to zero (0). To program the slave 82786, set the HSync/VSync (S) and Blank (B) Slave Mode bits in the CRTMode Register to one (1). For details on these bits, refer to Table 3-8 in Section 3.3.2 "Display Processor Control Block Registers." The HSync and VSync lines of the slave 82786 then become inputs and are driven by the HSync and VSync output lines of the master 82786. If the Window Status signals are used, the master's HSync and VSync signals should be qualified with the Blank signal (similar to Figure 6-7) to correctly drive the slave 82786 HSync

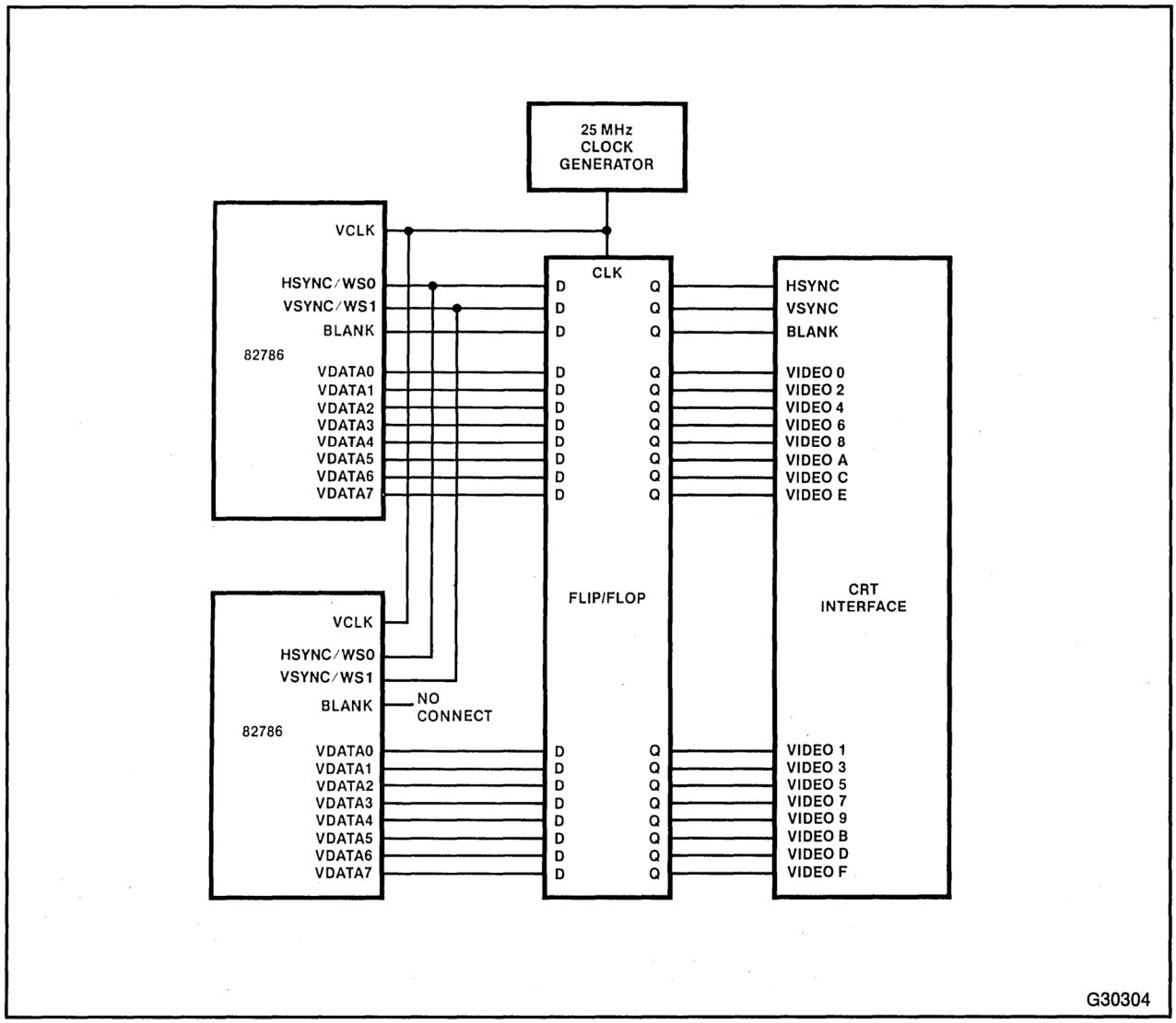


Figure 6-14. Dual 82786s Generate 16 Bpp at 25 MHz

and VSync inputs. If the external logic prevents Hsync and VSync from being driven while Blank is low, the slave 82786s may also enable their window status outputs. They will drive window status while Blank is low and sense HSync and VSync when Blank is high.

NOTE

In slave video mode, at least a 1-line vertical front porch and a 7-line vertical back porch are required.

The Video Timing Registers (HSyncStp, LineLen, VSyncStp, VFldStp, VFldStrt, FramLen) of both, master and slave 82786s should be programmed identically and HFldStrt and HFldStp on the slaves should be programmed to be 2 less than the the value of the Master. For details on these registers, refer to Table 3-8 in Section 3.3.2 "Display Control Block Registers." With these Video Timing Registers set, the slave 82786 automatically synchronizes itself to the master 82786 by waiting for its HSync input to rise before each scan line and waiting for its VSync input to rise before beginning a new display field. If a noninterlaced display is used, the two 82786s will always be synchronized.

6.5.1 An Interlaced Display with Two or More 82786s

If you interlace the display and use two 82786s, both 82786s should start on the same field, which can be done by ensuring they start scanning the display simultaneously. First set up the slave 82786 CRTMode and Video Timing Registers (HSyncStp, HFldStrt, HFldStp, LineLen, VSyncStp, VFldStp, VFldStrt, FramLen) with a LD_ALL command. This prepares the slave 82786 to begin scanning the display when HSync and VSync rise. HSync and VSync will be floating because they are tristated by all the 82786s. Then set up the master 82786 with a LD_ALL command to program its CRTMode and Video Timing Registers. After the master starts scanning, the HSync and VSync signals are driven by the master so both 82786s will begin on the even interlaced field.

6.5.2 Bitmap Configurations with Dual 82786s

To create a 16-bit per pixel (bpp) bitmap with two 82786s, each 82786 Graphics Processor (GP) is programmed for 8-bpp bitmaps of identical size. To draw in both bitmaps, a Graphic Command Block (GCMB) must be created for each 82786s. These GCMBs are generally identical for both 82786s except for the color values for the Def_Color command and the mask value for the Def_Logical_Op command. To display 16-bpp bitmaps, each 82786 is given an identical Strip Descriptor so each can display 8 bits of each 16-bit pixel.

Similarly, 8-bpp bitmaps can be created by splitting the bitmaps between both 82786s, with each 82786 responsible for four of the eight bpp. This splits the work between the two 82786s so that the Bit_Blt and Scan_Line fill graphics commands execute twice as fast. Also, with the Display Processor (DP) bus overhead split between the two 82786s, less bus contention exists, so all other drawing commands are faster.

Alternatively, 8-bpp bitmaps can be generated by just one of the 82786s. This minimizes the overhead between the host CPU and the 82786 because the CPU communicates with only one 82786.

The method in which the 16 video data bits are mapped into colors for the display interface determines which of the two preceding methods are used for bitmaps of 8 bpp or less. If the mapping is flexible enough, it may be feasible to create any bitmap depth. For example, 9-bpp bitmaps potentially can be created using one 82786 for eight bits and another for only one bit of each pixel.

The displays discussed in previous sections obtained high resolutions at the expense of bpp. By combining two or more 82786s, more bpp can be provided at these high resolutions.

Figure 6-15 shows a configuration that uses two 82786s to create a 4-bpp display at a video rate of 100 MHz. This configuration would support an 1144 × 860, 16-color, noninterlaced, 60 Hz display. Each 82786 generates two bits of each four bpp. Each 82786 draws and maintains half of the bitmap in its graphics memory; each 4-bpp window is divided into two 2-bpp bitmaps, with one 2-bpp bitmap generated by each 82786. Each Graphic Processor (GP) is programmed as normal for a 2-bpp bitmap. The Display Processors (DPs) are programmed as discussed in Section 6.5.1 "An Interlaced Display with Two or More 82786s." Each window is programmed with one-fourth the horizontal positioning resolution.

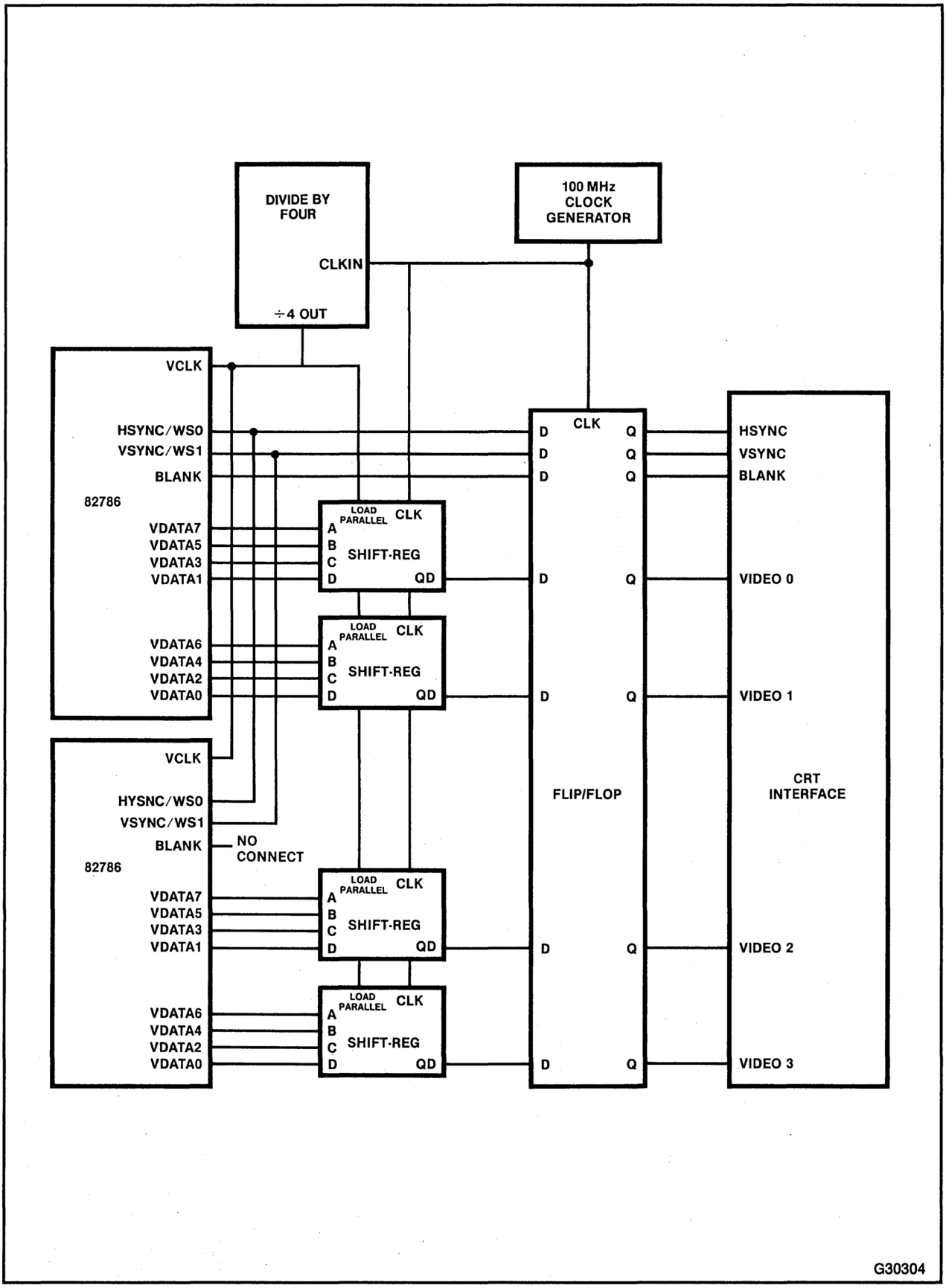


Figure 6-15. Two 82786s Create a 4-Bpp Display at 100 MHz

6.6 VIDEO RAM (VRAM) INTERFACE

The 82786 supports dual-port video DRAMs (VRAMs) to generate the video data stream. To enable the 82786 VRAM Mode, the VR bit in the Bus Interface Unit Control Register must be set to 1 (see Section 4.2.2 "Control Register"). In VRAM Mode, the first tile in each each Strip Descriptor generates VRAM Data Transfer (DT) cycles; the second tile and any subsequent tiles in the strip (up to 15) generate DRAM cycles. The pixel data for every scan line in the entire display must be contained in a single row in memory (256 words for noninterleaved memory and 512 words for interleaved memories). The Strip Descriptor for each tile indicates only 1 pixel with its address corresponding to the first display pixel. For details, refer to Section 3.2.5 "VRAM Support."

During the horizontal retrace period, the 82786 transfers the contents of the memory row containing the first pixel into the VRAM shift register. The VRAM shift clock is gated with a Blank signal. During the active display time, the shift clock is active and periodically clocks out the video data. External multiplexers must convert the 16-bit (32-bit interleaved) data stream into a serial stream based on the bits per pixel (bpp) needed as shown in Figure 6-15.

For the VRAM tile, the pixel depth is fixed in external hardware and all Display Processor (DP) Registers referring to video data fetch should be programmed to zero unless VRAM Hardware Overlay Mode is used (see Section 3.2.5.2 "Hardware Overlays"). When VRAM hardware overlays are implemented, the first Tile Descriptor is programmed for VRAM Mode and Tile Descriptors for the second tile and any subsequent tiles are programmed for DRAM Mode. For details, refer to Section 3.2.5 "VRAM Support."

6.7 EXTERNAL CHARACTER ROM

Few 82786 applications require, or even benefit from, using an external character ROM because the 82786 Graphic Processor (GP) very rapidly draws characters. The GP can fill an 80 × 25 character screen with highly detailed 16 × 16 characters in less than one tenth of a second. The GP also is very flexible in drawing characters; they can be:

- formed from an unlimited number of character fonts
- placed at any pixel on the screen
- rotated in four directions with four paths
- combined with graphics
- drawn in any color
- have a transparent or opaque background

Conversely, a character ROM display restricts the characters from a predefined font to character-cell positions on the screen with few, if any, of the above flexibilities.

However, for downward compatibility or support of an extremely large character set, implementing the character ROM function in an 82786 system may be practical. Figure 6-16 illustrates a system that displays both character ROM text and 82786 graphics. A multiplexer switches between the character ROM output and the direct 82786 output. One of the Window Status bits switches the multiplexer so both the character ROM and the 82786 graphics windows can be shown simultaneously on the same screen. The direct 82786 VDATA and Window Status signal must be delayed the same number of clocks as the character-cell video so that all signals get to the multiplexer on the same clock. The extra D-flip/flops before the multiplexer perform the needed delay.

The character ROM in Figure 6-16 can display 256 characters using a 9 × 14 pixel character-cell. Characters are stored as 8-bit pixels within an 82786 bitmap. To display a character, the window is programmed as an 8-bpp bitmap with a horizontal zoom of 9 and a vertical zoom of 14. The 82786 places the 8-bit character code on its VDATA pins during the scan lines when the character is displayed. The pixel counter loads the shift register every 9 pixels. When the Window Status pin falls, the pixel counter synchronizes to the beginning pixel of the window. The row counter supplies row information to the character ROM. This row counter synchronizes to the frame by starting from the end of the VSync pulse, which means that all character ROM windows in this example must begin at a multiple of 14 scan lines after VSync.

The Japanese Kanji character set is a good example in which character ROM support is practical. The size of this character set is so large that storing the characters in a character ROM is more practical than loading characters from disk into the 82786 graphics memory. Figure 6-17 illustrates a configuration that can display up to 65,536 characters from a very detailed (32 × 32) font. The circuit in Figure 6-17 allows both text and graphics windows to be displayed on the screen simultaneously. One of the window status signals selects between text and graphics.

A character set such as Kanji requires high resolution, and is generally achieved using a monochrome display. The circuit in Figure 6-17 allows up to 200 MHz video at one bpp for very high resolution screens.

The 82786 is programmed in Super High-Speed Acceleration Mode (see Section 6.4.4 "Examples of High Resolution Configurations"). The character codes to be displayed should be placed in 1-bpp bitmaps with 16 consecutive bits for each character. The hardware combines the 8-bit VDATA values from two consecutive pixels to generate the 16-bit character code for the Character ROM. If less than 65,536 characters are required, not all of the 16-bit character code addresses need to be used for the character ROM. Some of these bits can be used for attributes such as blinking and reverse video. The ROM contains a 32 × 32 character font, each character is split into 32-lines of four 8-bit bytes. The "pane" counter selects one of the four 8-bit bytes at a time. The "row" counter determines the current row of the character. Character cell windows are zoomed by 2 horizontally and by 32 vertically. The window must be placed at a multiple of 4-pixels from HSync and a multiple of 32-lines from VSync. It is possible to place windows at non-multiples from HSync and VSync if the "pane" and "row" counters' parallel inputs are tied to signals other than ground.

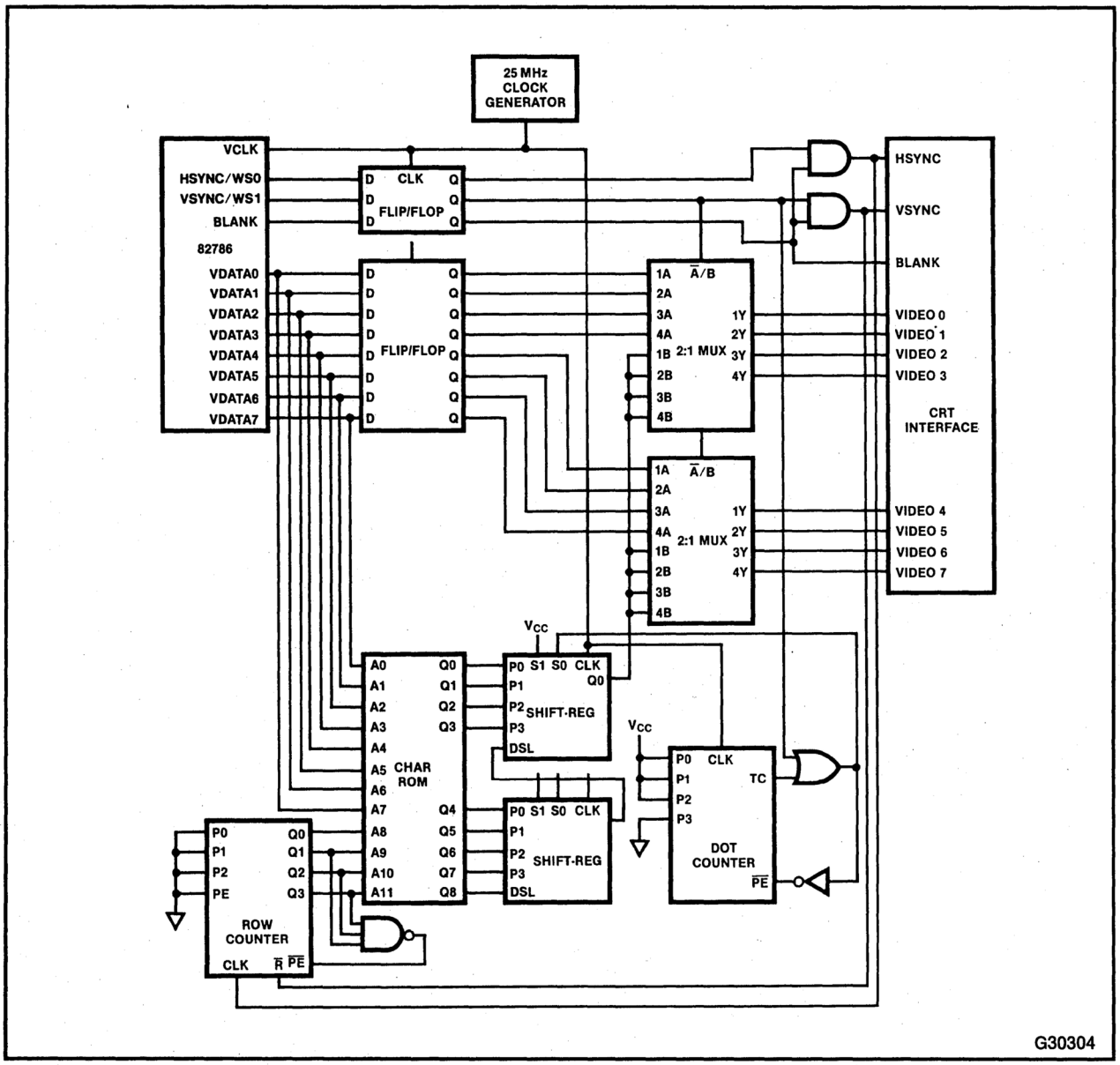


Figure 6-16. Character ROM and Bitmap Graphics on Same Screen

6.8 COMBINING THE 82786 WITH OTHER VIDEO SOURCES

Graphic output from the 82786 can be combined with output from other video sources such as broadcast TV, video recorders, and video laser disc players. The main requirement is that both the 82786 and the video source are locked in synchronization.

The 82786 supports a Master and two independent Video Slave Modes. In Master Mode, HSync, VSync, and Blank are outputs. When HSync and VSync are outputs, they still are inputs during the active display period if the Window Status bits are enabled (see WSt in Table 3-1 of Section 3.1.3.2 "Strip Descriptor Format" and the W bit in the CrtMode Register in Table 3-8 of Section 3.3.2 "Display Control Block Registers"). In both Slave Modes, HSync and VSync are inputs while Blank can be either an input or an output. External HSync and VSync reset the 82786 horizontal and vertical counters respectively.

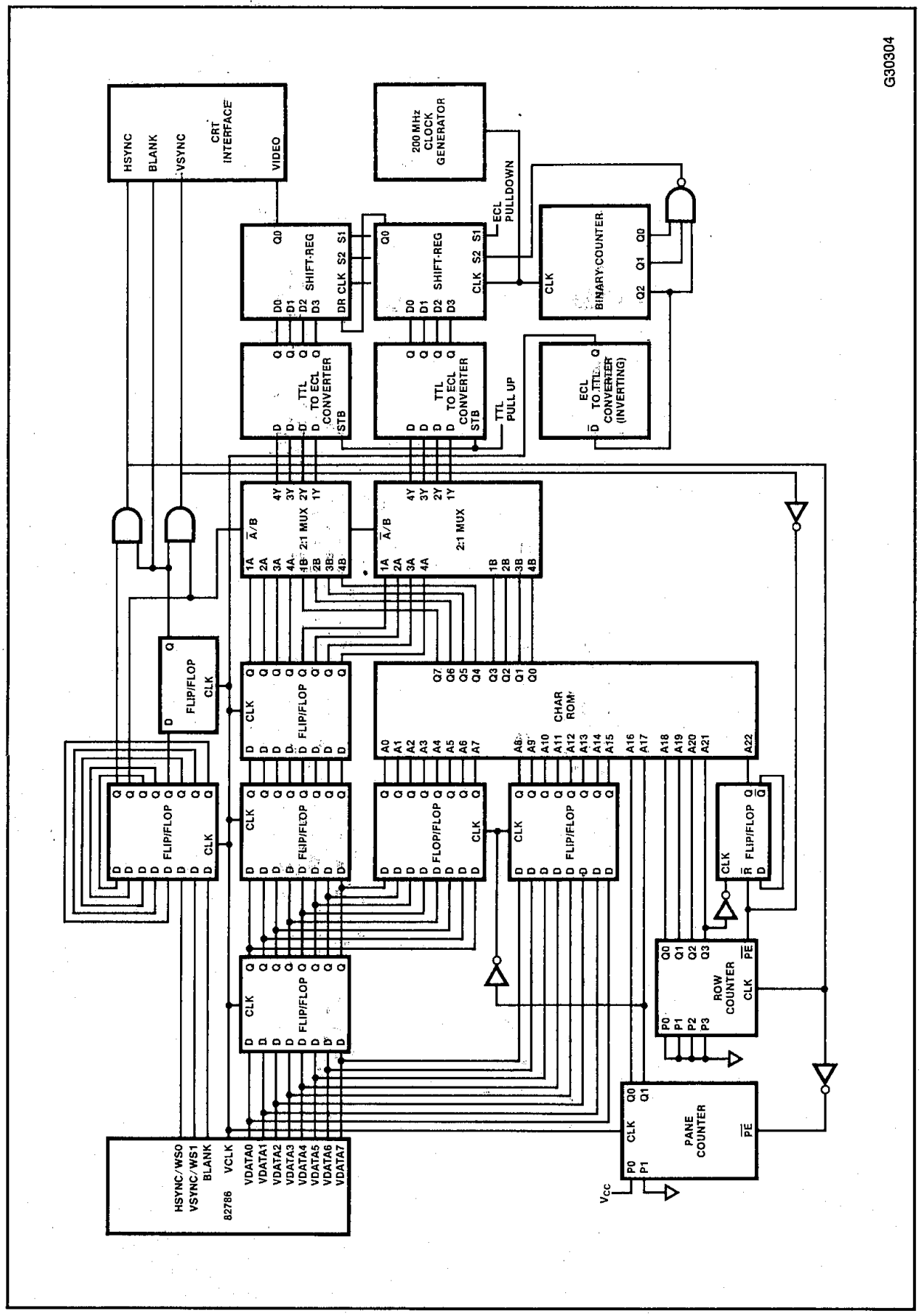


Figure 6-17. Very Large Character ROM and Bitmap Graphics on Same Screen

The programming of Blank determines whether the active display period is determined by the 82786 Video Timing Registers or the external video source. When Blank is configured as an output, the programmed values of the Video Timing Registers VFldStrt, VFldStp, HFldStrt, and HFldStp (see Table 3-8 in Section 3.3.2 "Display Control Block Registers") determine the active display period. When Blank is configured as an input, the external system determines the active display period. The internal video shift register generates video data only during the active display period. Table 6-4 outlines the normal programming states for HSync, VSync, and Blank.

The 82786 should be programmed for Slave Video Synchronization. The synchronization signals from the video source must be converted into separate TTL-level HSync and VSync and fed to the 82786 HSync and VSync pins. The 82786 then automatically synchronizes itself to the video source by waiting for its HSync input to fall before each scan line and waiting for its VSync input to fall before beginning a new display field.

For many applications, the 82786 Video Clock (VClk) can be derived directly from a crystal oscillator. Because the 82786 synchronizes to the nearest pixel on every scan line, even video sources with imperfect timings produce an acceptable picture. The frame-to-frame deviation of the 82786 graphics information on the screen relative to the video source is never more than one pixel.

For more demanding applications, the 82786 VClk can be synthesized directly from the video source timings using a phase-locked-loop circuit. The 82786 still synchronizes itself every scan line, but the relationship between HSync and the 82786 VClk remains constant. This implementation creates virtually no deviation between the 82786 graphics and the video source.

For interlaced video, the 82786 display must initially start just prior to the VSync before an even field. The 82786 initialization software must guarantee that the first LD_ALL to start the 82786 display occurs sufficiently before the VSync during an odd field so the first 82786 display field matches the video source even field.

After the 82786 is locked in synchronization with the video source, the VDATA information from the 82786 is easily combined with the video from the video source. Although the two video signals can be mixed on top of each other, the most common implementation is to switch the two sources. With the Window Status bits to initiate the switching, the 82786

Table 6-4. HSync, VSync, and Blank Settings

HSync and VSync	Blank	Application
Output	Output	Master/Stand-alone display generated by 82786
Input	Output or Input	82786 generated display superimposed on externally generated video or Slave 82786s in a multiple 82786 system

can create letters that are placed over the video picture. During the display scan, whenever a portion of a letter is to be displayed, the video from the 82786 can be switched in, otherwise the video source will be switched in.

If the output of the video source is analog, the 82786 VDATA can be converted into an analog signal and an analog switch can be used. The state of the switch can be derived several ways. If the switching is to be done on window boundaries, one Window Status pin can control the switch. If the switching must be done within a window, a special graphics color code can indicate to replace the 82786 video with that from the video source. For example, place the color 11111111 on the VDATA pins with an 8-input NAND gate to control the analog switch.

6.9 OTHER TYPES OF DISPLAYS AND PRINTERS

In addition to CRTs, the 82786 supports other types of displays including LCD, plasma, and intelligent printers. These devices have diverse interface requirements, which cannot be discussed in detail here, but following sections outline basic concepts for most device interfaces.

6.9.1 Pixel Clock Rate

The rate at which pixels are clocked into displays can vary immensely. The 82786 allows a wide range of VClk frequencies from DC levels to 25 MHz that accommodate many devices. In addition, faster clock rates can be generated using the method described in Section 6.6 "Video RAM (VRAM) Interface."

No Refresh Printers and some displays do not require continuous refresh. Needlessly running the 82786 Display Processor (DP) through refresh cycles steals bus bandwidth from the Graphic Processor (GP) and other processors. To eliminate this waste, the display can be turned off by resetting the DspOn bit (bit 0) in the DP Video Status (VStat) Register (see Table 3-8 in Section 3.2.2 "Display Control Block Registers"). When DspOn is reset, the DP continues to generate HSync, VSync, and Blank and place Default VDATA on the VDATA lines, but no bus bandwidth is required. When a change to the display is required, the DspOn bit can be set using the LD_REG or LD_ALL command. After the refresh starts, another LD_REG command can be placed in the DP Opcode Register to turn the display back off (see Section 3.3.1 "Display Processor Internal Registers"). The DP then automatically executes it when the refresh is completed. For details on the LD_REG or LD_ALL commands, refer to Sections 3.6.1 and 3.6.2.

6.9.2 Partial Display Updates

Some displays that do not require continuous refresh have a long update time. To update every pixel on the display may take several seconds. For small changes to the display, such as adding a character as it is typed by the user, updating only the affected portion of the display may be more feasible than updating the entire display. With the windowing capability of the 82786, an application can scan through a specific portion of the display.

6.9.3 Pixel Address Generation

Some displays, especially those which allow only partial display updates, require pixel location addresses to be generated along with the pixel data. Although external circuitry can generate these addresses, the 82786 also can generate them directly. If a single 8-bit address is all that is required, program the Display Processor (DP) Default Video Register to a value so the VDATA pins can output it during Blank time (see Section 3.3.1 "Display Processor Internal Registers"). With proper programming of the Video Timing Registers, this value can be clocked into the display before each scan line using HSync. For details on the Video Timing Registers, refer to HSyncStp, HFldStrt, HFldStp, LineLen, VSyncStp, VFldStp, VFldStrt, FramLen in Table 3-8 of Section 3.3.2 "Display Control Block Registers."

More complex pixel addresses can be generated by using the 82786 windowing capability. Create a thin window at the beginning of each scan line, and one or more bytes of address information can be clocked out sequentially over the VDATA pins before each line.

6.9.4 Super High Resolution

With some displays that use either slow refresh times or do not require refresh, very high resolutions are possible. All of the display counters in the 82786 are 12 bits, which allows up to a 4096 × 4096 display size (some of this resolution may not be available depending on the number of synchronization clock cycles required). By trading off bit per pixel (bpp) for resolution, the Accelerated Modes can provide 2, 4, or 8 times this resolution horizontally, up to 32,768 pixels (see Section 6.4.1 "External Logic Requirements").

Some applications, such as printers, require greater resolutions. Greater resolution can be achieved by the 82786 using external counters to generate the HSync, VSync, and Blank inputs for the 82786. Program the 82786 Slave Video Mode by setting the CRTMode Register Slave parameters (see the S and B bits in Table 3-8 of Section 3.3.2 "Display Control Block Registers"). With all 16 horizontal windows, the horizontal resolution can be up to 4096 × 16 = 65,536 pixels. Trading bpp for resolution, the Accelerated Modes can provide 2, 4, or 8 times this resolution, up to 524,288 pixels. Vertical resolution has no limit other than that imposed by available memory.

Such high resolutions require a large amount of memory. For example, a printer generating 300 × 300 dots per square inch on 8.5 × 11 inch paper at only one bpp (no gray scale) requires the following amount of memory to print the entire page.

$$\frac{300 \times 300 \times 8.5 \times 11}{8 \text{ bits}} = 1.05 \text{ Megabytes}$$

Placing this much memory in a printer may not be feasible, but memory requirements can be decreased by printing the display one strip at a time. Suppose the first 300 lines are generated and printed. After they are printed, the next 300 lines can be generated and printed, using the same memory, as shown in Figure 6-18.

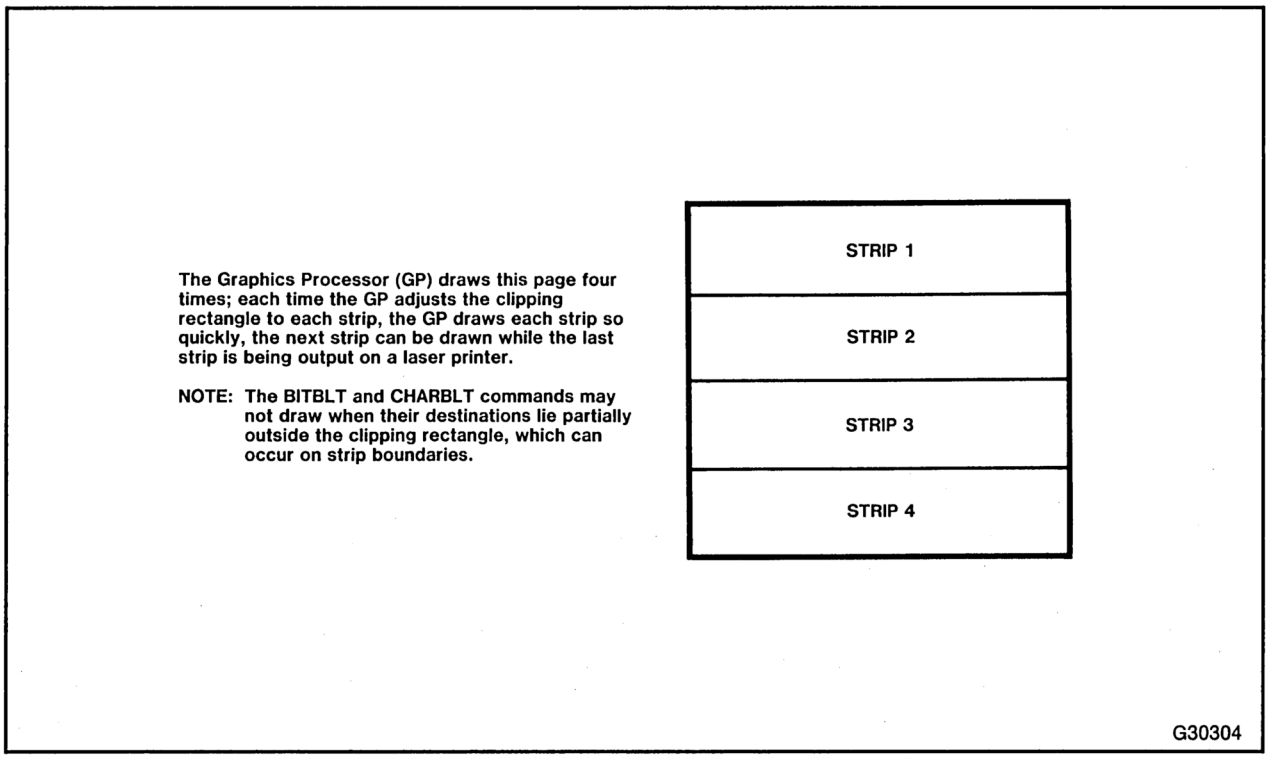


Figure 6-18. Dividing a Page into Strips Decreases Memory Requirements

By printing the display in 300-line strips, the memory required is only:

$$\frac{300 \times 300 \times 8.5}{8 \text{ bits}} = 96 \text{ Kbytes}$$

If the image printed can be described by a set of commands for the Graphic Processor (GP), each strip can be generated very easily. The drawing bitmap and clipping rectangle are set for the first strip and the GP then runs through the Graphic Command Block (GCMB). After completing the strip, it can be printed. The bitmap and clipping rectangle are set for the second strip. The GP traverses the same command block again to generate the second strip, and so on, until the page is printed.

If enough memory exists for two strips, double buffering can pipeline the operation. While the Display Processor (DP) prints one strip, the GP generates the next strip. The same approach can be extended to multiple pages, even using more than one 82786.

6.10 CALCULATING VIDEO PARAMETERS

The 82786 video Display Processor (DP) is programmable and provides a wide variety of display formats. To determine the display format(s), application, monitor, and video interface parameters must be considered and calculated. The following subsections discuss some of these parameters.

6.10.1 Application Parameters

Application parameters, outlined below, are based on the needs of the specific application and must be chosen by the designer.

Hres, Horizontal resolution
Hres is the number of pixels per horizontal line. When using Accelerated Video Modes, Hres must be a multiple of the 82786 Video Acceleration (Accel) described in Section 6.10.3 "Video Interface Parameters."

Vres, Vertical resolution
Vres is the number of vertical pixels (scan lines) per display.

Vfreq, Vertical frequency
Vfreq is the rate at which the CRT beam makes one pass from the top of the screen to the bottom. Often 60 Hz is used, but almost any frequency can be generated by the 82786. The U.S. broadcast television standards use a 59.95 Hz rate. European video systems are based on 50 Hz field rate. High resolution displays often use 40 Hz or lower. Slower rates reduce the speed requirements of the monitor and the 82786 video circuitry, which also permits higher resolutions. However, slower rates also flicker more and may be intolerable to the operator. Generally, rates significantly under 60 Hz will tend to cause some perceptible flicker unless CRTs with long persistence phosphor are used.

IL, Interlacing
IL is an interlaced display that uses two consecutive fields to generate a frame. A noninterlaced display generates the entire display frame in one field scan. Interlacing doubles the resolution of a display. Figure 6-19 shows noninterlaced and interlaced displays.

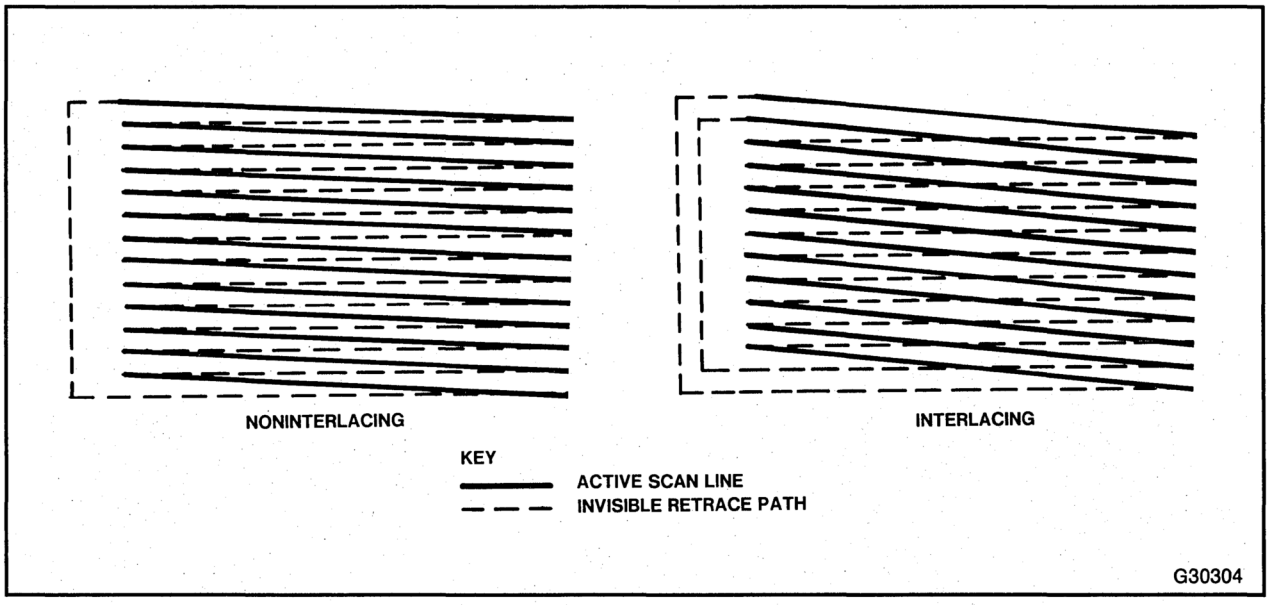


Figure 6-19. Noninterlaced and Interlaced Displays

In an interlaced display, alternate scan lines are written during each alternate field. For TV-like pictures, where the image generally doesn't change drastically from one line to the next, interlacing allows a 30 Hz frame rate with a 60 Hz field rate without perceptible flicker. For detailed computer graphics, however, one line may change drastically from the next in color and/or intensity, which causes perceptible flicker at such rates.

The 82786 supports both noninterlaced and interlaced displays. In addition, an interlaced-sync mode is available, which generates synchronization signals in the manner used by interlaced displays, but generates the video signals in the manner used by noninterlaced displays (both fields identical). This permits interlaced screens with consecutive pairs of identical lines.

6.10.2 Monitor Parameters

The monitor parameters outlined below are based on the requirements of the display monitor used.

HBlank, Horizontal Blanking time
HBlank is the time required for the CRT beam to move from the right side of the display back to the left and stabilize. This is also called horizontal retrace time. It is the sum of the horizontal synchronization (HSync) and front and back porch times as shown in Figure 6-20. Monitors typically range from 5 to 12 μS.

VBlank, Vertical Blanking time
VBlank is the time required for the CRT beam to jump from the bottom of the display back to the top and stabilize. This is also called vertical retrace time. It is the sum of the vertical synchronization and front and back porch times as shown in Figure 6-21. Monitors typically range from 600 to 1400 μS.

Hfreq, Horizontal frequency
Hfreq is the frequency at which horizontal lines are scanned. Monitors typically range from 15 to 36 kHz.

Vfreq, Vertical frequency
Vfreq is the frequency at which the display field is scanned. Monitors typically range from 40 to 70 Hz.

BPP, bits per pixel
BPP is the number of bits assigned to each pixel. Monitors with digital inputs restrict the number of usable bpp. Monitors with analog inputs allow a virtually unlimited range of intensities with the use of Digital-to-Analog converters. This parameter is mainly dependent on the video interface hardware described in previous sections.

Color monitors generally limit the perceivable horizontal and vertical resolution due to their shadow mask. See the specific monitor specifications for more details.

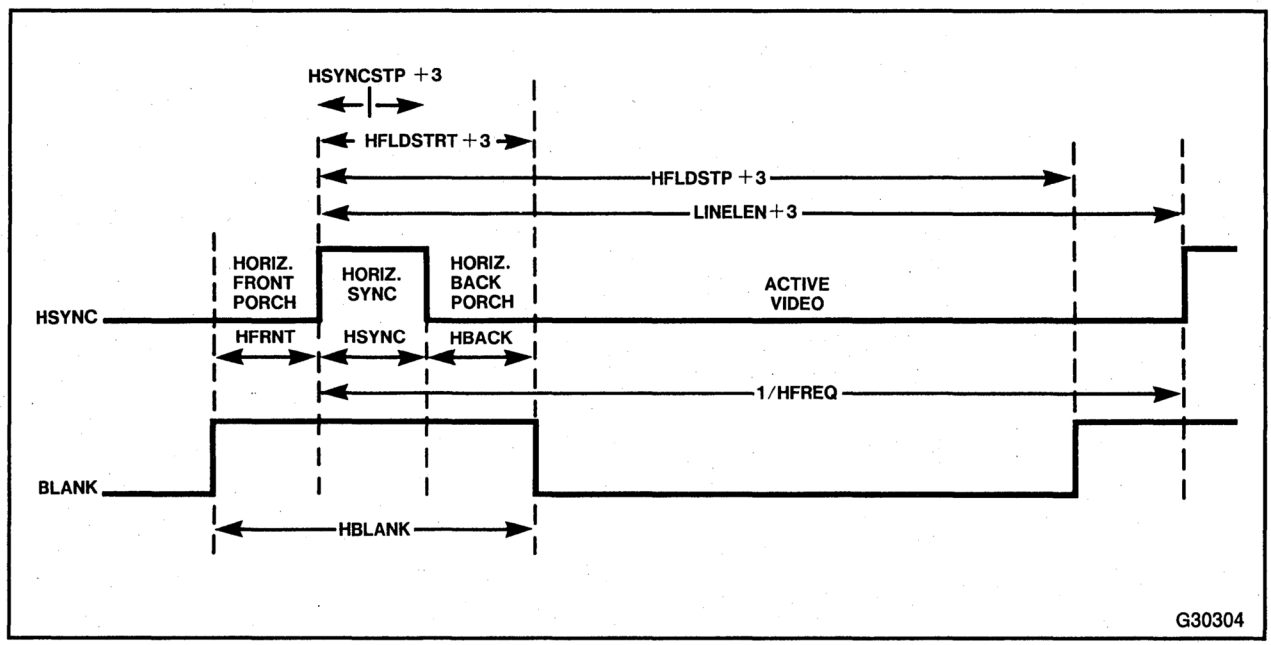


Figure 6-20. HSync and Blank Timing Parameters

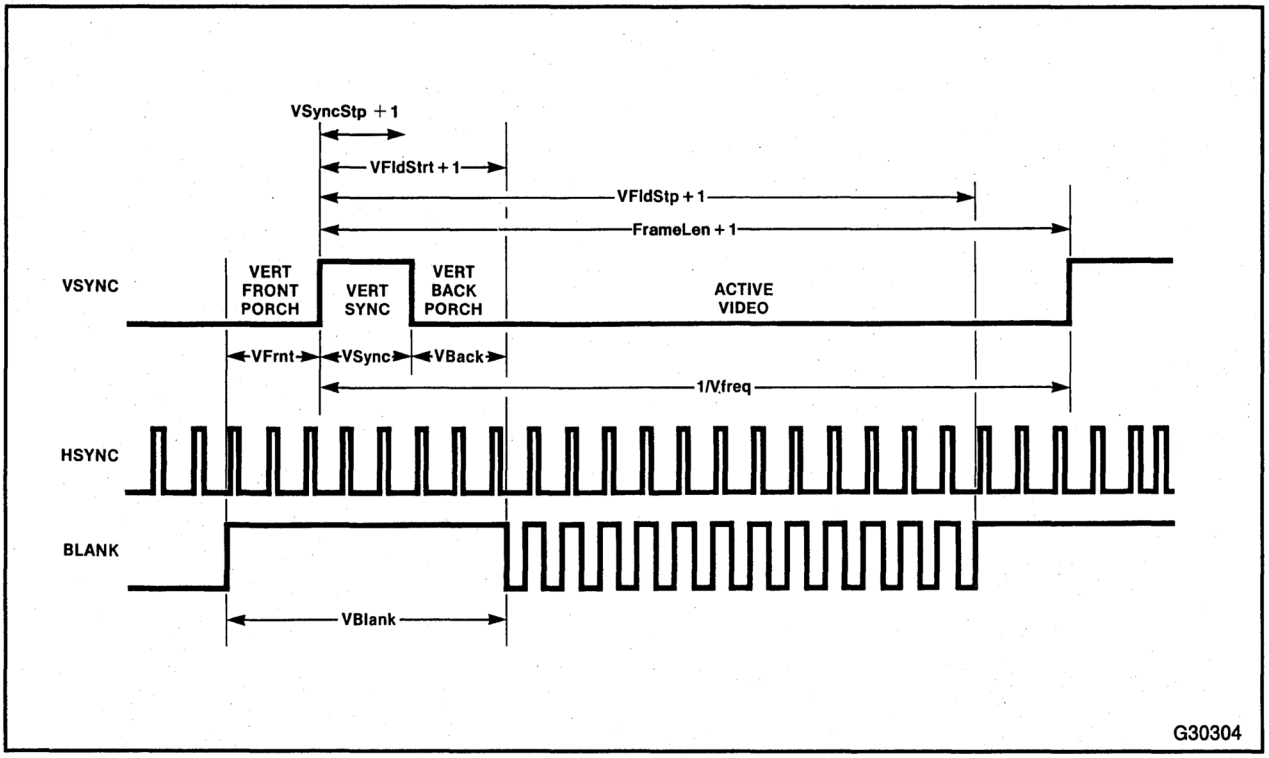


Figure 6-21. VSync and Blank Timing Parameters

6.10.3 Video Interface Parameters

The video interface parameters outlined below are dependent on the 82786 component and the video interface logic.

VClk, Video clock frequency
VClk is the video input clock into the 82786. It has a maximum rate of 25 MHz and may be chosen so that the frequency evenly divides by both Hfreq and Vfreq.

Accel, 82786 video acceleration
Accel is determined by the mode in which the 82786 operates. Normally, Accel equals 1. If the bit per pixel (bpp) tradeoffs mentioned in Section 6.4 "High Resolutions with Standard DRAMs" are used to attain higher video rates at the expense of fewer bpp, then the value for Accel can be 2, 4, or 8 as shown in Table 6-5.

DotClk, Pixel dot clock frequency
DotClk is normally the same as VClk. However, when Accelerated Video Modes are used, DotClk is either 2, 4, or 8 times VClk.

$$DotClk = VClk \times Accel$$

HSyncStp, HFldStrt, HFldStp, LineLen
These Video Timing values are programmed into the 82786 Display Processor Display Control Block to determine the horizontal scan timing in Figure 6-20. They may be set to any value from 0 to 4095. Their values should also fit the formula:

$$HSyncStp < HFldStrt < HFldStp < LineLen$$

VSyncStp, VFldStrt, VFldStp, FramLen
These Video Timing values are programmed into the 82786 Display Processor (DP) to determine the vertical scan timing in Figure 6-22. They may be set to any value from 0 to 4095. Their values should also fit the formula:

$$VSyncStp < VFldStrt < VFldStp < FramLen$$

After the preceding parameters are evaluated, the video parameters can be calculated. The parameters interact quite heavily. For example, if a specific horizontal and vertical resolution at a specific field rate is required, the monitor frequencies and Blank times may need to be altered. Several iterations may be necessary to optimize all the parameters.

Table 6-5. Accelerated Video Mode Values

Video Mode	Max DotClk	Programmed Accel Bits
Accel=1 Normal	25 MHz	0 0
Accel=2 High-Speed	50 MHz	0 1
Accel=4 Very High-Speed	100 MHz	1 0
Accel=8 Super High-Speed	200 MHz	1 1

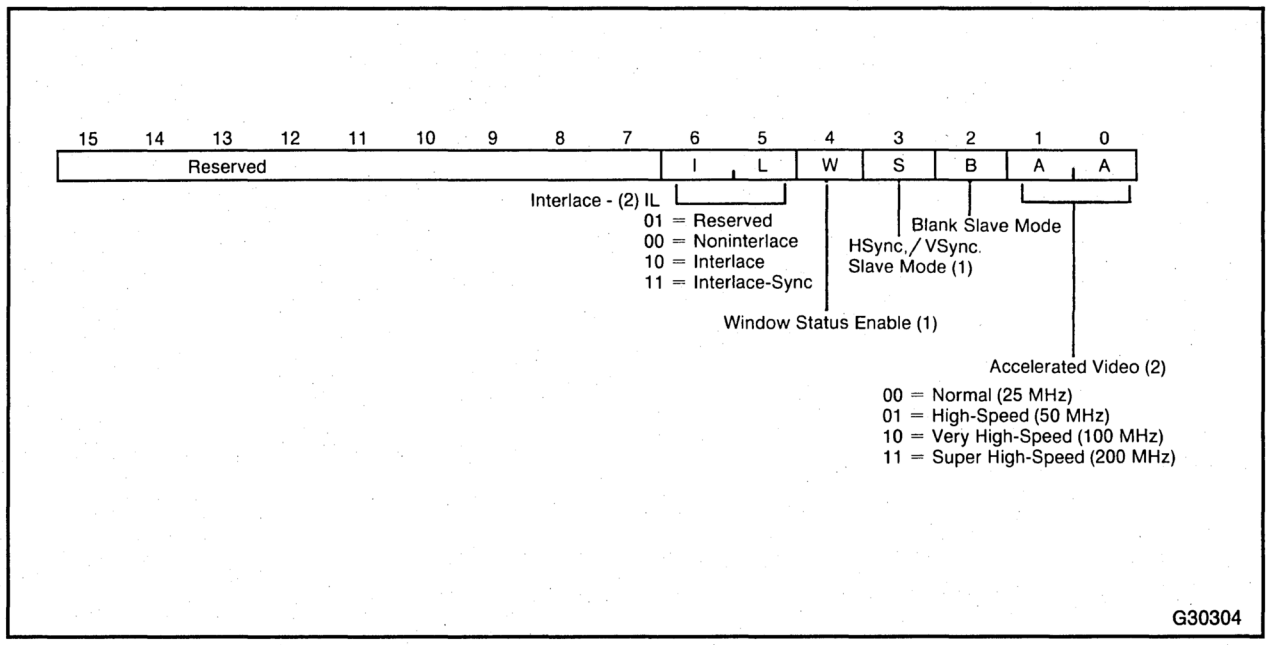


Figure 6-22. Display Processor Register 05H

6.10.4 Sample Video Parameter Calculations

The following formulas allow you to determine the video parameters. A sample calculation is included for your convenience. For the example, generate a 640 × 400 × 8 bpp (256 color) screen at 60 Hz noninterlaced. Assume:

Hres = 640 pixels
Vres = 400 pixels
Vfreq% = 60 Hz
Hblank% = 12 μS
VBlank% = 1300 μS
Accel = 1 (no external acceleration)

Variables with a percent (%) after them represent desired values, the actual value are calculated below. ROUND(X) denotes rounding off X to the nearest integer.

1. Start by calculating the vertical resolution per field. Since the display is noninterlaced, the value is the same as the vertical resolution. If the display were interlaced then:

VresFld = Vres/2
else: VresFld = Vres VresFld
= 400 pixels

With interlaced screens, VresFld is half the vertical resolution. For example, with 525 lines, use 262.5 for VresFld.

2. Now calculate the horizontal frequency required. Subtract the Vertical Blank time from the vertical period and divide by the active vertical lines to get the horizontal period. Invert this to get the horizontal frequency.

$$\text{Hfreq\%} = \frac{1}{\text{Hperiod\%}} = \frac{\text{VresFld}}{(1/\text{Vfreq\%}) - \text{VBlank\%}}$$

$$= \frac{400}{(1/60) - 1300\ \mu\text{S}} = 26.03\text{kHz}$$

3. Calculate the pixel dot clock required.

$$\text{DotClk\%} = \frac{1}{\text{DotPeriod\%}} = \frac{\text{Hres}}{(1/\text{Hfreq\%}) - \text{HBlank\%}}$$

$$= \frac{640}{(1/26.03\ \text{kHz}) - 12\ \mu\text{S}} = 24.23\ \text{MHz}$$

4. Calculate the actual 82786 VClk. Because the example does not use external acceleration circuits VClk is the same as the DotClk.

$$\text{VClk\%} = \text{DotClk\%/Accel}$$
$$= 24.23\ \text{MHz}/1$$
$$= 24.23\ \text{MHz}$$

5. The example requires a 24.23 MHz crystal to generate VClk. Such a crystal is difficult to locate, so rework the example using a more standard 25 MHz crystal. See how it affects the rest of the parameters. First, the pixel dot clock changes.

$$\text{DotClk} = \text{VClk} \times \text{Accel}$$
$$= 25.00\ \text{MHz} \times 1$$
$$= 25.00\ \text{MHz}$$

6. Next, determine how many VClks are required for the horizontal time.

$$\text{HBlankClks} = \text{ROUND}(\ \text{VClk} \times \text{Hblank\%}\ )$$
$$= \text{ROUND}(\ 25\ \text{MHz} \times 12\ \mu\text{S})$$
$$= 300$$

7. Now, calculate the actual Horizontal Blank time.

$$\text{Hblank} = \frac{\text{HBlankClks}}{\text{VClk}} = \frac{300}{25\ \text{MHz}} = 12\ \mu\text{S}$$

The actual horizontal perioa is the time required to display one line of pixels plus the Blanking time.

$$\text{Hfreq} = \frac{1}{(\text{Hres/DotClk}) + \text{HBlank}}$$

$$= \frac{1}{(640/25\ \text{MHz}) + 12\ \mu\text{S}}$$

8. The number of horizontal lines per frame can now be calculated:

VFrameLines = ROUND(Hfreq / Vfreq%)
= ROUND(26.60 kHz / 60 Hz)
= 443

If an interlaced display is used, VFrameLines should be rounded-off to the closest odd integer.

9. The number of scan lines determines the actual vertical frequency:

Vfreq = Hfreq / VFrameLines
= 26.60 kHz / 443
= 60.05 Hz

10. With the major parameters calculated, the Blanking times can be broken into synchronization (sync) and front and back porch times. Typical monitor values are:

HSync = 2 μS VSync = 300 μS
HBack = 6 μS VBack = 800 μS

HSyncClks = ROUND(VClk × HSync)
= ROUND(25 MHz × 2 μS)
= 50

HBackClks = ROUND(VClk × HBack)
= ROUND(25 MHz × 6 μS)
= 150

VSyncClks = ROUND(Hfreq × VSync)
= ROUND(26.6 kHz × 300 μS)
= 8

VBackClks = ROUND(Hfreq × VBack)
= ROUND(26.6 kHz × 800 μS)
= 21

11. Now calculate the values for the eight 82786 Display Processor Video Timing Registers.

HSyncStp = HSyncClks − 3
= 50 − 3
= 47

HFldStrt = HSyncStp + HBackClks
= 47 + 150
= 197

HFldStp = HFldStrt + (Hres / Accel)
= 197 + (640 / 1)
= 837

LineLen = HBlankClks + (Hres / Accel) − 3
= 300 + (640 / 1) − 3
= 937

For noninterlaced displays:

VSyncStp = VSyncLines − 1
= 8 − 1
= 7

VFldStrt = VSyncStp + VBackLines
= 7 + 21
= 28

VFldStp = VFldStrt + Vres
= 28 + 400
= 428

FrameLen = VFrameLines − 1
= 443 − 1
= 442

For interlaced and interlace-sync displays:

VSyncStp = (VSyncLines × 2) − 1
VFldStrt = VSyncStp + (VBackLines × 2)
VFldStp = VFldStrt + (VRes ÷ 2)
FrameLen = VFldStp + (2 × VFrontLines)

In the preceding formula, VResTotal equals the Vertical Resolution for Field1 plus the Vertical Resolution for Field2 as shown below.

VResTotal = VResField1 + VResField2

VFrameLines equals the total number of HSyncs in an entire frame. LineLen should be greater than HFldStp. FrameLen should be greater than VFrameLines. If not, the parameters are inconsistent and the requirements should be modified and recalculated.

Finally, the bits for the CRTMode Register should be determined. In this example, noninterlaced mode is used and no Accelerated Video is required. Assuming the 82786 generates the HSync, VSync, and Blank signals; and assuming the Window Status pins are not used, the CRTMode Registers should be loaded with all zeroes.

The host CPU software must load the values of the eight Video Timing Registers (HSyncStp, HFldStrt, HFldStp, LineLen, VSyncStp, VFldStp, VFldStrt, FrameLen and the CRTMode Register (see Table 3-8 in Section 3.3.2 "Display Control Block Registers"). Generally, this is done during system initialization. All registers should be loaded simultaneously using the LD_ALL instruction rather than using individual LD_REG instructions to ensure that the video synchronization signals (HSync, VSync, and Blank) are never invalid while registers are being loaded.

Some CRTs can be damaged permanently by supplying the wrong synchronization frequencies to them. To prevent invalid video sync signals, the HSync, VSync, and Blank pins are tristated after RESET until the CRTMode Register has been initialized.

6.11 A SPREADSHEET FOR CALCULATING VIDEO PARAMETERS

Section 6.10.4 illustrates that determining the 82786 video parameter constants requires many calculations. Often, optimizing the display format requires several iterations through the calculations. This time consuming and painstaking process can be simplified by using a spreadsheet. An example of output from a spreadsheet follows. The example illustrates a 1290 × 968 × 4 bpp (16 color), interlaced, 60 Hz display. The "DESIRED" column contains user supplied values and the spreadsheet calculates all other values. The "ACTUAL" column shows the closest timings and parameters that the 82786 can supply. The "82786 DP REGISTER VALUES" show the values that should be programmed into the Display Processor Registers to generate such a display.

The user easily can modify the "DESIRED" values until the "ACTUAL" values meet the application's needs. Ensure that all "ACTUAL" values are logically correct. For example, if any of the calculated parameters are negative, then the set of "DESIRED" parameters cannot produce such a display, so some parameters must be adjusted.

82786 VIDEO PARAMETERS
Type under DESIRED column only: ACTUAL & REGISTER columns are calculated

PARAMETER	DESIRED	ACTUAL	DP REGISTER VALUES	
Video Clock VClk (MHz):	25	25		
Acceleration (1,2,4 or 8):	2	2		
Interlacing (1=no, 2=yes):	2	2		
Horiz Resolution (Pixels):	1290	1290		
Vert. Resolution (Pixels):	968	968		
Horiz Line Rate (kHz):	---------	30.487	LineLen:	818
Horiz Sync Width (uS):	2	2	HSyncStp:	8
Horiz Back Porch (uS):	4	4	HFldStrt:	148
Horiz Front Porch (uS):	1	1	HFldStp:	793
Vert. Frame Rate (Hz):	60	59.956	FrameLen:	1015
Vert. Sync Width (uS):	200	196.8	VSyncStp:	10
Vert. Back Porch (uS):	400	393.6	VFldStrt:	34
Vert. Front Porch (uS):	---------	213.2	VFldStp:	1002

The template used for this spreadsheet follows. The template should adapt easily to nearly any spreadsheet program. This particular spreadsheet program uses @ROUND(X,0) to denote rounding to the nearest integer. If no rounding function is available in your spreadsheet program, you can substitute the integer function (which truncates the fractional portion to return the next lowest integer) for the round function:

substitute @INT(X+0.5) for @ROUND(X,0)

After entering the template into your favorite spreadsheet, you can verify that it is working correctly by entering the "DESIRED" values of the above example and checking that the "ACTUAL"and "REGISTER" results match.

	A		B	C	D	E
1:		**82786 Video Parameters**				
2:		**Type under DESIRED column only; ACTUAL & REGISTER columns are calculated**				
3:						
4:	**PARAMETER**		**DESIRED**	**ACTUAL**	**82786 DP REGISTER VALUES**	
5:						
6:	Video Clock VCLK	(MHz):		+B6		
7:	Acceleration	(1,2,4 or 8):		+B7		
8:	Interlacing	(1=no, 2=yes):		+B8		
9:	Horiz. Resolution	(Pixels):		@ROUND(B9/C7,0)*C7		
10:	Vert. Resolution	(Pixels):		@ROUND(B10,0)		
11:						
12:	Horiz. Line Rate	(kHz):	---	(C6*1000)/(E12+2)	LineLen:	@ROUND(C6*B15,0)+E15
13:	Horiz. Sync Width	(uS):		(E13+2)/C6	HSyncStp:	@ROUND(C6*B13,0)−3
14:	Horiz. Back Porch	(uS):		(E14-E13)/C6	HFldStrt:	@ROUND(C6*B14,0)+E13
15:	Horiz. Front Porch	(uS):		(E12-E15)/C6	HfldStp:	+E14+(C9/C7)
16:						
17:	Vert. Frame Rate	(Hz):		(C8*C12*1000)/(E17+C8)	FrameLen:	@ROUND((C12*1000)/B17−(C8−1)/2,0)*C8−1
18:	Vert. Sync Width	(uS):		((E18+C8)*1000)/(C12*C8)	VsyncStp:	(@Round((C12*B18)/1000,0)−1)*C8
19:	Vert. Back Porch	(uS):		((E19-E18)*1000)/(C12*C8)	VFldStrt:	@ROUND((C12*B19)/1000,0)−1)*C+E18
20:	Vert. Front Porch	(uS):	---	(E17-E20)*1000)/(C12*C8)	VFldStp:	+E19+C10

Test Modes

A

APPENDIX A
TEST MODES

The 82786 has three test modes that help in debugging, characterization, and production testing. The 82786 test modes are activated when the 82786 implements any of three global pin conditioning features, which allow the 82786 to drive all output and I/O pins high, low, or tristate all pins. When RESET goes inactive, the 82786 samples the Read ($\overline{RD}$) and Write ($\overline{WR}$) pins. If either of these pins is low, one of the test modes is enabled according to the state of $\overline{RD}$, $\overline{WR}$, and MIO pins as outlined in Table A-1.

Table A-1. 82786 Test Modes

$\overline{RD}$	$\overline{WR}$	MIO	Mode
0	0	0	DRAM Test Mode
0	0	1	Reserved
0	1	0	Reserved
0	1	1	Drive output pins high
1	0	0	Drive output pins low
1	0	1	Tristate pins
1	1	X	Normal operation

A.1 VOLTAGE OUT LOW (VOL) AND VOLTAGE OUT HIGH (VOH) PIN CONDITIONING

The 82786 can bring all its output pins to a constant logic high or low state, which can help test the 82786 output buffers.

A.2 TRISTATE FEATURE

The 82786 can tristate all of its I/O and output pins to effectively isolate the 82786 from any connected circuitry. This isolates the 82786 and allows testing of a completely assembled PC board. Leakage on all I/O pins can also be tested in this mode.

A.3 DRAM TEST MODE

The RESET input does not initialize the Refresh Scaler in the BIU DRAM/VRAM Control Register (see Section 4.2.3), which allows RESET input to be a warm reset without losing the contents of memory. From a testing point of view, this results in problems of synchronizing the tester with the 82786 Refresh Scaler. To overcome this, the DRAM test mode initializes the BIU DRAM/VRAM Control Register, but has no other effect.

88-Pin Grid Array

B

APPENDIX B
88-PIN GRID ARRAY

Figure B-1 shows the top view and Figure B-2 shows the bottom view pin configuration of the 82786. Table B-1 describes the function of each pin.

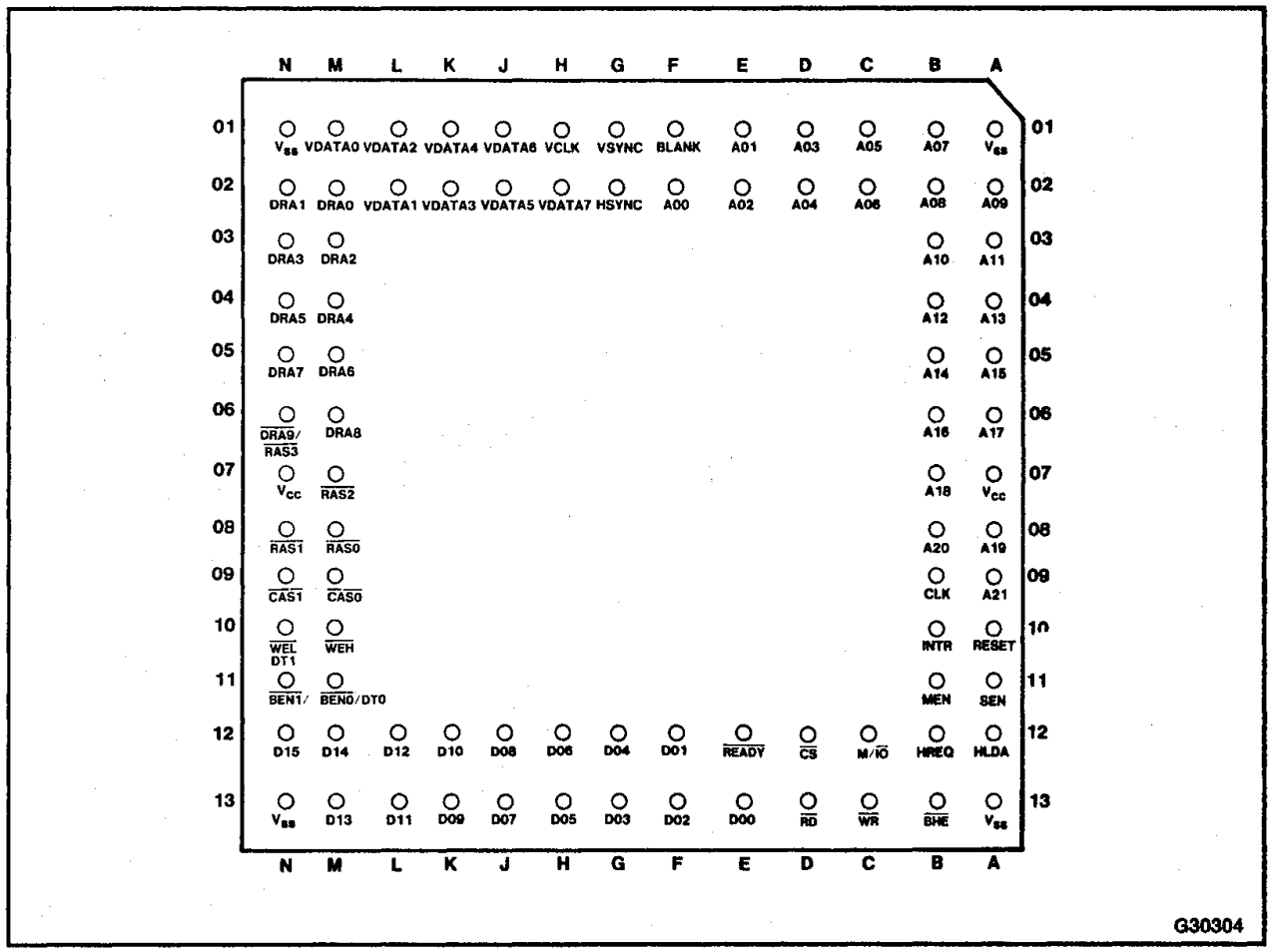


Figure B-1. 82786 Pinout—Top View

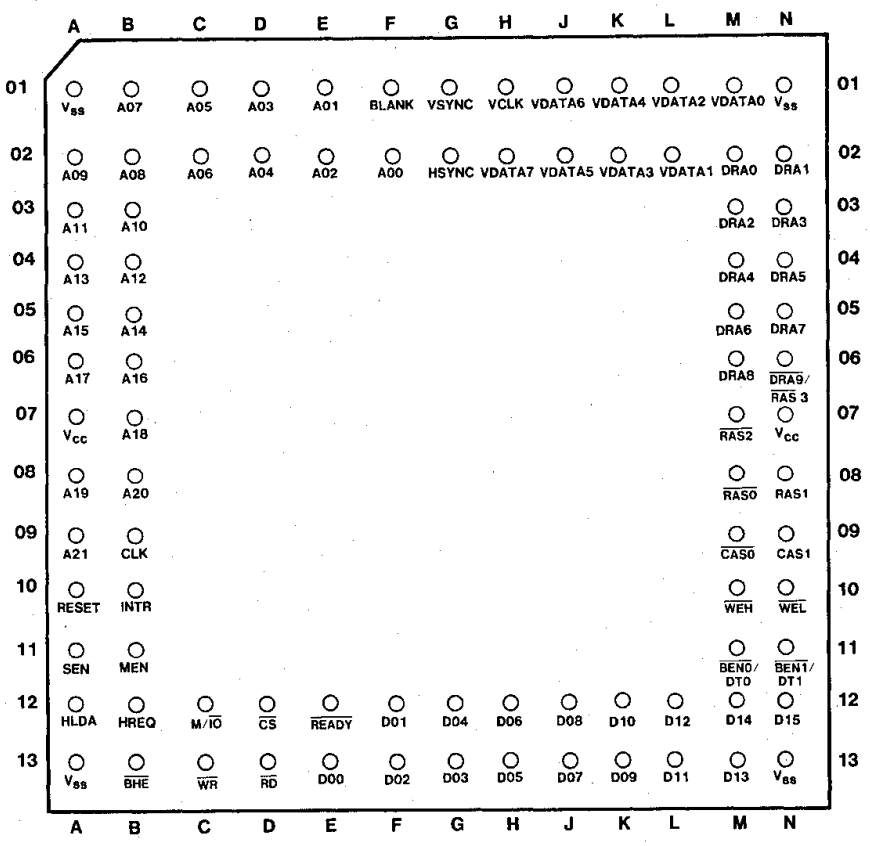


Figure B-2. Pinout 82786—Bottom View

Table B-1. Pin Descriptions

Symbol	Pin(s)	Type	Description
A21:0	A09,B08,A08, B07,A06,B06,A05 B05,A04,B04,A03, B03,A02,B02,B01, C02,C01,D02,D01, E02,E01,F02	I/O	Address lines for the External Bus Normally inputs for Slave accesses to 82786 graphics memory array or 82786 memory or I/O mapped internal registers. 82786 drives these lines when it is the Bus Master.
D15:0	N12,M12,M13,L12, L13,K12,K13,J12, J13,H12,H13,G12, G13,F13,F12,E13	I/O	Data Bus: For 82786 graphics memory array and External Bus.
$\overline{BHE}$	B13	I/O	Bus High Enable: An 82786 Slave input. 82786 drives it low when in Master Mode. $\overline{BHE}$ determines asynchronous or synchronous operation for $\overline{RD}$, $\overline{WR}$, and HLDA at falling (trailing) edge of RESET. A HIGH state selects synchronous operation.
$\overline{RD}$	D13	I/O	Read Strobe: An 82786 Slave input. 82786 drives $\overline{RD}$ when 82786 is Bus Master. Selects normal/test mode at falling RESET.

Table B-1. Pin Descriptions (Cont'd.)

Symbol	Pin(s)	Type	Description
$\overline{WR}$	C13	I/O	Write Strobe: An 82786 Slave input; 82786 drives $\overline{WR}$ when 82786 is Bus Master. Selects normal/ test mode at falling RESET.
M/$\overline{IO}$	C12	I/O	Memory or I/O Indication: 82786 Slave input. 82786 drives it high when the 82786 is the Bus Master.
$\overline{CS}$	D12	I	Chip Select: Slave input qualifies an access to graphics memory or internal registers.
MEN	B11	O	Master Enable: Driven high when the 82786 is the External Bus Master (i.e., HLDA received in response to an 82786 HREQ). Used to steer the data path and select source of bus cycle status commands.
SEN	A11	O	Slave Enable: Driven high when the 82786 executes a Slave bus cycle for an External Bus Master accessing graphics memory or internal registers. Used to enable the data path and indicate READY to the External Bus Master.
$\overline{READY}$	E12	I	Synchronous Input: to 82786 when executing external bus cycles. Identical to 80286 $\overline{READY}$ timing.
HREQ	B12	O	Hold Request: 82786 drives it high when the Display or Graphics Processor accesses the External Bus. Remains high until the 82786 no longer needs the External Bus.
HLDA	A12	I	Hold Acknowledge: Input in response to a HREQ output. Asynchronous or synchronous input determined by state of $\overline{BHE}$ pin at falling RESET.
INTR	B10	O	Interrupt: The logical OR of a Graphics Processor and Display Processor interrupt. An access to the BIU Control Register clears the interrupt.
RESET	A10	I	Reset Input: Internally synchronized, halts all 82786 activity and brings it to a defined state. The leading edge of RESET synchronizes the 82786 clock to 01. The trailing edge latches the state of $\overline{BHE}$ to establish a synchronous or asynchronous Slave interface. It also latches $\overline{RD}$, $\overline{WR}$, and MIO to set certain test modes, described in Appendix A.
CLK	B09	I	Double frequency clock input. Clock input to which pin timings are referenced. 50% duty cycle.

Table B-1. Pin Descriptions (Cont'd.)

Symbol	Pin(s)	Type	Description
$\overline{CAS0}$	M09	O	Column Address Strobe 0: Drives the CAS inputs of the even word graphics memory bank if interleaved; identical to $\overline{CAS1}$ if noninterleaved graphics memory. Can drive 16 DRAM CAS inputs.
$\overline{CAS1}$	N09	O	Column Address Strobe 1: Drives the CAS inputs of the odd word graphics memory bank if interleaved; identical to $\overline{CAS0}$ if noninterleaved graphics memory. Can drive 16 DRAM CAS inputs.
$\overline{RAS2:0}$	M07,N08,M08	O	Row Address Strobe: Drives the RAS input pins of up to 16 DRAMs. Drives the first three rows of both banks of graphics memory.
DRA9/ $\overline{RAS3}$	N06	O	Multiplexes Most Significant graphics memory Address Line & $\overline{RAS3}$: DRA9 when 1 M DRAMs are used; $\overline{RAS3}$ otherwise.
$\overline{WEL}$	N10	O	Write Enable Low Byte: Active LOW strobe to the low order byte of graphics memory.
$\overline{WEH}$	M10	O	Write Enable High Byte: Active LOW strobe to the high order byte of graphics memory.
DRA8:0	M06,N05,M05, N04,M04,N03, M03,N02,M02	O	Multiplexed Graphics Memory Address: Graphics memory row and column address are multiplexed on these lines. Can directly drive 32 DRAMs.
$\overline{BEN1}$:0/ DT1:0	N11,M11	O	Multiplexed Bank Enable and Data Transfer Line: For memory rad/write cycles, enables graphics memory array output on the 82786 data bus, D15:0. In a VRAM data transfer cycle, loads the shift register in dual port VRAMs. $\overline{BEN1}$/DT1 controls Bank 1 and $\overline{BEN0}$/DT0 controls Bank 0.
BLANK	F01	I/O	Output Blanks the Display during horizontal and vertical retrace. Can be configured as input to synchronize the 82786 with external sources.
VDATA 7:0	H02,J01,J02, K01,K02,L01, L02,M01	O	Video Data Output.
VCLK	H01	I	Video Clock Input: Drives the 82786 display section. Maximum frequency of 25 MHz.
HSYNC/ WSt0	G02	I/O	Horizontal Synchronization and multiplexed window status. Can be configured as input to synchronize the 82786 with external sources. Also can be configured to output Window Status. See Section 3.2.3 "HSync and VSync Multiplexes Window Status."

Table B-1. Pin Descriptions (Cont'd.)

Symbol	Pin(s)	Type	Description
VSYNC/ WSt1	G01	I/O	Vertical Synchronization and multiplexed window status. Can be configured as inputs to synchronize the 82786 with external sources. Also can be configured to output Window Status. See Section 3.2.3 "HSync and VSync Multiplexes Window Status."
V_{SS}	A01,N01,A13,N13		4 V_{SS} Pins. Ground
V_{CC}	N07,A07		2 V_{CC} Pins. +5 Volts

82786 Commands

C

APPENDIX C
82786 COMMANDS

GRAPHICS PROCESSOR COMMANDS

Command	Opcode (Higher Byte - Hex)
Absolute Move (Abs_Mov)	4Fh
Arc_Exclusion	68h
Arc_Inclusion	69h
Bit_Blt	64h
BitBlt_M	AEh
Call	0Fh
Char_Opaque	A6h
Char_Transparent	A7h
Circle	8Eh
Def Bitmap	1Ah
Def Char_Set_Byte	0Bh
Def Char_Set_Word	0Ah
Def Char_Space	4Dh
Def Char_Orient	4Eh
Def Clip_Rect	46h
Def Color	3Dh
Def Logical_Op	41h
Def Texture_Opaque	06h
Def Texture_Transparent	07h
Dump_Reg	29h
Enter_Pick	44h
Exit Pick	45h
Halt	xx01
Incr_Point	B4h
Intr_Gen	0Eh
Line	54h
Link	02h
Load_Reg	34h
NOP	03h
Point	53h
Polygon	73h
PolyLine	74h
Rect	58h
Rel_Move	52h
Return	17h
Scan_Lines	BAh

DISPLAY PROCESSOR COMMANDS

Command	Opcode (Higher Byte - Hex)
Load Register (LD_Reg)	04h
Load All (LD_All)	05h
Dump Register (Dmp_Reg)	06h
Dump All (Dmp_All)	07h

DOMESTIC SALES OFFICES

ALABAMA

†Intel Corp.
5015 Bradford Dr., #2
Huntsville 35805
Tel: (205) 830-4010

ARIZONA

†Intel Corp.
11225 N. 28th Dr.
Suite D-214
Phoenix 85029
Tel: (602) 869-4980

†Intel Corp.
1161 N. El Dorado Place
Suite 301
Tucson 85715
Tel: (602) 299-6815

CALIFORNIA

†Intel Corp.
21515 Vanowen Street
Suite 116
Canoga Park 91303
Tel: (818) 704-8500

†Intel Corp.
2250 E. Imperial Highway
Suite 218
El Segundo 90245
Tel: (213) 640-6040

†Intel Corp.
1510 Arden Way, Suite 101
Sacramento 95815
Tel: (916) 920-8096

†Intel Corp.
4350 Executive Drive
Suite 105
San Diego 92121
Tel: (619) 452-5880

†Intel Corp.*
400 N. Tustin Avenue
Suite 450
Santa Ana 92705
Tel: (714) 835-9642
TWX: 910-595-1114

†Intel Corp.*
San Tomas 4
2700 San Tomas Expressway
2nd Floor
Santa Clara 95051
Tel: (408) 986-8086
TWX: 910-338-0255
FAX: 408-727-2620

COLORADO

†Intel Corp.
4445 Northpark Drive
Suite 100
Colorado Springs 80907
Tel: (719) 594-6622

†Intel Corp.*
650 S. Cherry St., Suite 915
Denver 80222
Tel: (303) 321-8086
TWX: 910-931-2289

CONNECTICUT

†Intel Corp.
26 Mill Plain Road
2nd Floor
Danbury 06811
Tel: (203) 748-3130
TWX: 710-456-1199

FLORIDA

†Intel Corp.
6363 N.W. 6th Way, Suite 100
Ft. Lauderdale 33309
Tel: (305) 771-0600
TWX: 510-956-9407
FAX: 305-772-8193

†Intel Corp.
5850 T.G. Lee Blvd.
Suite 340
Orlando 32822
Tel: (305) 240-8000
FAX: 305-240-8097

†Intel Corp.
11300 4th Street North
Suite 170
St. Petersburg 33716
Tel: (813) 577-2413
FAX: 813-578-1607

GEORGIA

†Intel Corp.
3280 Pointe Parkway
Suite 200
Norcross 30092
Tel: (404) 449-0541

ILLINOIS

†Intel Corp.*
300 N. Martingale Road, Suite 400
Schaumburg 60173
Tel: (312) 310-8031

INDIANA

†Intel Corp.
8777 Purdue Road
Suite 125
Indianapolis 46268
Tel: (317) 875-0623

IOWA

Intel Corp.
1930 St. Andrews Drive N.E.
2nd Floor
Cedar Rapids 52402
Tel: (319) 393-5510

KANSAS

†Intel Corp.
10985 Cody St.
Suite 140, Bldg. D
Overland Park 66210
Tel: (913) 345-2727

MARYLAND

†Intel Corp.*
7321 Parkway Drive South
Suite C
Hanover 21076
Tel: (301) 796-7500
TWX: 710-862-1944

†Intel Corp.
7833 Walker Drive
Suite 550
Greenbelt 20770
Tel: (301) 441-1020

MASSACHUSETTS

†Intel Corp.*
Westford Corp. Center
3 Carlisle Road
2nd Floor
Westford 01886
Tel: (617) 692-3222
TWX: 710-343-6333

MICHIGAN

†Intel Corp.
7071 Orchard Lake Road
Suite 100
West Bloomfield 48322
Tel: (313) 851-8096

MINNESOTA

†Intel Corp.
3500 W. 80th St., Suite 360
Bloomington 55431
Tel: (612) 835-6722
TWX: 910-576-2867

MISSOURI

†Intel Corp.
4203 Earth City Expressway
Suite 131
Earth City 63045
Tel: (314) 291-1990

NEW JERSEY

†Intel Corp.*
Parkway 109 Office Center
328 Newman Springs Road
Red Bank 07701
Tel: (201) 747-2233

†Intel Corp.
280 Corporate Center
75 Livingston Avenue
First Floor
Roseland 07068
Tel: (201) 740-0111
FAX: 201-740-0626

NEW MEXICO

†Intel Corp.
8500 Menaul Boulevard N.E.
Suite B 295
Albuquerque 87112
Tel: (505) 292-8086

NEW YORK

Intel Corp.
127 Main Street
Binghamton 13905
Tel: (607) 773-0337
FAX: 607-723-2677

†Intel Corp.*
850 Cross Keys Office Park
Fairport 14450
Tel: (716) 425-2750
TWX: 510-253-7391

†Intel Corp.*
2950 Expressway Dr., South
Suite 130
Islandia 11722
Tel: (516) 231-3300
TWX: 510-227-6236

†Intel Corp.
Westage Business Center
Bldg. 300, Route 9
Fishkill 12524
Tel: (914) 897-3860
FAX: 914-897-3125

NORTH CAROLINA

†Intel Corp.
5700 Executive Drive
Suite 213
Charlotte 28212
Tel: (704) 568-8966

†Intel Corp.
2700 Wycliff Road
Suite 102
Raleigh 27607
Tel: (919) 781-8022

OHIO

†Intel Corp.*
3401 Park Center Drive
Suite 220
Dayton 45414
Tel: (513) 890-5350
TWX: 810-450-2528

†Intel Corp.*
25700 Science Park Dr., Suite 100
Beachwood 44122
Tel: (216) 464-2736
TWX: 810-427-9298

OKLAHOMA

†Intel Corp.
6801 N. Broadway
Suite 115
Oklahoma City 73162
Tel: (405) 848-8086

OREGON

†Intel Corp.
15254 N.W. Greenbrier Parkway
Building B
Beaverton 97006
Tel: (503) 645-8051
TWX: 910-467-8741

PENNSYLVANIA

†Intel Corp.*
455 Pennsylvania Avenue
Suite 230
Fort Washington 19034
Tel: (215) 641-1000
TWX: 510-661-2077

Intel Corp.*
400 Penn Center Blvd., Suite 610
Pittsburgh 15235
Tel: (412) 823-4970

PUERTO RICO

†Intel Microprocessor Corp.
South Industrial Park
P.O. Box 910
Las Piedras 00671
Tel: (809) 733-8616

TEXAS

†Intel Corp.
313 E. Anderson Lane
Suite 314
Austin 78752
Tel: (512) 454-3628

†Intel Corp.*
12000 Ford Road
Suite 400
Dallas 75234
Tel: (214) 241-8087
FAX: 214-484-1180

†Intel Corp.*
7322 S.W. Freeway
Suite 1490
Houston 77074
Tel: (713) 988-8086
TWX: 910-881-2490

UTAH

†Intel Corp.
428 East 6400 South
Suite 104
Murray 84107
Tel: (801) 263-8051

VIRGINIA

†Intel Corp.
1504 Santa Rosa Road
Suite 108
Richmond 23288
Tel: (804) 282-5668

WASHINGTON

†Intel Corp.
155 108th Avenue N.E.
Suite 386
Bellevue 98004
Tel: (206) 453-8086
TWX: 910-443-3002

†Intel Corp.
408 N. Mullan Road
Suite 102
Spokane 99206
Tel: (509) 928-8086

WISCONSIN

†Intel Corp.
330 S. Executive Dr.
Suite 102
Brookfield 53005
Tel: (414) 784-8087
FAX: (414) 796-2115

CANADA

BRITISH COLUMBIA

Intel Semiconductor of Canada, Ltd.
4585 Canada Way, Suite 202
Burnaby V5G 4L6
Tel: (604) 298-0387
FAX: (604) 298-8234

ONTARIO

†Intel Semiconductor of Canada, Ltd.
2650 Queensview Drive
Suite 250
Ottawa K2B 8H6
Tel: (613) 829-9714
TLX: 053-4115

†Intel Semiconductor of Canada, Ltd.
190 Attwell Drive
Suite 500
Rexdale M9W 6H8
Tel: (416) 675-2105
TLX: 06983574
FAX: (416) 675-2438

QUEBEC

†Intel Semiconductor of Canada, Ltd.
620 St. John Boulevard
Pointe Claire H9R 3K2
Tel: (514) 694-9130
TWX: 514-694-9134

†Sales and Service Office
*Field Application Location

DOMESTIC DISTRIBUTORS

ALABAMA

Arrow Electronics, Inc.
1015 Henderson Road
Huntsville 35805
Tel: (205) 837-6955

†Hamilton/Avnet Electronics
4940 Research Drive
Huntsville 35805
Tel: (205) 837-7210
TWX: 810-726-2162

Pioneer/Technologies Group, Inc.
4825 University Square
Huntsville 35805
Tel: (205) 837-9300
TWX: 810-726-2197

ARIZONA

†Hamilton/Avnet Electronics
505 S. Madison Drive
Tempe 85281
Tel: (602) 231-5140
TWX: 910-950-0077

Hamilton/Avnet Electronics
30 South McKiemy
Chandler 85226
Tel: (602) 961-6669
TWX: 910-950-0077

Arrow Electronics, Inc.
4134 E. Wood Street
Phoenix 85040
Tel: (602) 437-0750
TWX: 910-951-1550

Wyle Distribution Group
17855 N. Black Canyon Hwy.
Phoenix 85023
Tel: (602) 249-2232
TWX: 910-951-4282

CALIFORNIA

Arrow Electronics, Inc.
10824 Hope Street
Cypress 90630
Tel: (714) 220-6300

Arrow Electronics, Inc.
19748 Dearborn Street
Chatsworth 91311
Tel: (213) 701-7500
TWX: 910-493-2086

†Arow Electronics, Inc.
521 Weddell Drive
Sunnyvale 94086
Tel: (408) 745-6600
TWX: 910-339-9371

Arrow Electronics, Inc.
9511 Ridgehaven Court
San Diego 92123
Tel: (619) 565-4800
TWX: 888-064

†Arrow Electronics, Inc.
2961 Dow Avenue
Tustin 92680
Tel: (714) 838-5422
TWX: 910-595-2860

†Avnet Electronics
350 McCormick Avenue
Costa Mesa 92626
Tel: (714) 754-6071
TWX: 910-595-1928

†Hamilton/Avnet Electronics
1175 Bordeaux Drive
Sunnyvale 94086
Tel: (408) 743-3300
TWX: 910-339-9332

†Hamilton/Avnet Electronics
4545 Ridgeview Avenue
San Diego 92123
Tel: (619) 571-7500
TWX: 910-595-2638

†Hamilton/Avnet Electronics
9650 Desoto Avenue
Chatsworth 91311
Tel: (818) 700-1161

†Hamilton Electro Sales
10950 W. Washington Blvd.
Culver City 20230
Tel: (213) 558-2458
TWX: 910-340-6364

Hamilton Electro Sales
1361B West 190th Street
Gardena 90248
Tel: (213) 217-6700

†Hamilton/Avnet Electronics
3002 'G' Street
Ontario 91761
Tel: (714) 989-9411

†Avnet Electronics
20501 Plummer
Chatsworth 91351
Tel: (213) 700-6271
TWX: 910-494-2207

CALIFORNIA (Cont'd.)

†Hamilton Electro Sales
3170 Pullman Street
Costa Mesa 92626
Tel: (714) 641-4150
TWX: 910-595-2638

†Hamilton/Avnet Electronics
4103 Northgate Blvd.
Sacramento 95834
Tel: (916) 920-3150

Wyle Distribution Group
124 Maryland Street
El Segundo 90254
Tel: (213) 322-8100

Wyle Distribution Group
7382 Lampson Ave.
Garden Grove 92641
Tel: (714) 891-1717
TWX: 910-348-7140 or 7111

Wyle Distribution Group
11151 Sun Center Drive
Rancho Cordova 95670
Tel: (916) 638-5282

†Wyle Distribution Group
9525 Chesapeake Drive
San Diego 92123
Tel: (619) 565-9171
TWX: 910-335-1590

†Wyle Distribution Group
3000 Bowers Avenue
Santa Clara 95051
Tel: (408) 727-2500
TWX: 910-338-0296

†Wyle Distribution Group
17872 Cowan Avenue
Irvine 92714
Tel: (714) 863-9953
TWX: 910-595-1572

Wyle Distribution Group
26677 W. Agoura Rd.
Calabasas 91302
Tel: (818) 880-9000
TWX: 372-0232

COLORADO

Arrow Electronics, Inc.
7060 South Tucson Way
Englewood 80112
Tel: (303) 790-4444

†Hamilton/Avnet Electronics
8765 E. Orchard Road
Suite 708
Englewood 80111
Tel: (303) 740-1017
TWX: 910-935-0787

†Wyle Distribution Group
451 E. 124th Avenue
Thornton 80241
Tel: (303) 457-9953
TWX: 910-936-0770

CONNECTICUT

†Arrow Electronics, Inc.
12 Beaumont Road
Wallingford 06492
Tel: (203) 265-7741
TWX: 710-476-0162

Hamilton/Avnet Electronics
Commerce Industrial Park
Commerce Drive
Danbury 06810
Tel: (203) 797-2800
TWX: 710-456-9974

†Pioneer Electronics
112 Main Street
Norwalk 06851
Tel: (203) 853-1515
TWX: 710-468-3373

FLORIDA

†Arrow Electronics, Inc.
400 Fairway Drive
Suite 102
Deerfield Beach 33441
Tel: (305) 429-8200
TWX: 510-955-9456

Arrow Electronics, Inc.
37 Skyline Drive
Suite 3101
Lake Mary 32746
Tel: (407) 323-0252
TWX: 510-959-6337

†Hamilton/Avnet Electronics
6801 N.W. 15th Way
Ft. Lauderdale 33309
Tel: (305) 971-2900
TWX: 510-956-3097

†Hamilton/Avnet Electronics
3197 Tech Drive North
St. Petersburg 33702
Tel: (813) 576-3930
TWX: 810-863-0374

FLORIDA (Cont'd.)

†Hamilton/Avnet Electronics
6947 University Boulevard
Winter Park 32792
Tel: (305) 628-3888
TWX: 810-853-0322

†Pioneer/Technologies Group, Inc.
337 S. Lake Blvd.
Alta Monte Springs 32701
Tel: (407) 834-9090
TWX: 810-853-0284

Pioneer/Technologies Group, Inc.
674 S. Military Trail
Deerfield Beach 33442
Tel: (305) 428-8877
TWX: 510-955-9653

GEORGIA

†Arrow Electronics, Inc.
3155 Northwoods Parkway
Suite A
Norcross 30071
Tel: (404) 449-8252
TWX: 810-766-0439

†Hamilton/Avnet Electronics
5825 D Peachtree Corners
Norcross 30092
Tel: (404) 447-7500
TWX: 810-766-0432

Pioneer/Technologies Group, Inc.
3100 F Northwoods Place
Norcross 30071
Tel: (404) 448-1711
TWX: 810-766-4515

ILLINOIS

Arrow Electronics, Inc.
1140 W. Thorndale
Itasca 60143
Tel: (312) 250-0500
TWX: 312-250-0916

†Hamilton/Avnet Electronics
1130 Thorndale Avenue
Bensenville 60106
Tel: (312) 860-7780
TWX: 910-227-0060

MTI Systems Sales
1100 W. Thorndale
Itasca 60143
Tel: (312) 773-2300

†Pioneer Electronics
1551 Carmen Drive
Elk Grove Village 60007
Tel: (312) 437-9680
TWX: 910-222-1834

INDIANA

†Arrow Electronics, Inc.
2495 Directors Row, Suite H
Indianapolis 46241
Tel: (317) 243-9353
TWX: 810-341-3119

Hamilton/Avnet Electronics
485 Gradle Drive
Carmel 46032
Tel: (317) 844-9333
TWX: 810-260-3966

†Pioneer Electronics
6408 Castleplace Drive
Indianapolis 46250
Tel: (317) 849-7300
TWX: 810-260-1794

IOWA

Hamilton/Avnet Electronics
915 33rd Avenue, S.W.
Cedar Rapids 52404
Tel: (319) 362-4757

KANSAS

Arrow Electronics
8208 Melrose Dr., Suite 210
Lenexa 66214
Tel: (913) 541-9542

†Hamilton/Avnet Electronics
9219 Quivera Road
Overland Park 66215
Tel: (913) 888-8900
TWX: 910-743-0005

Pioneer/Tec Gr.
10551 Lockman Rd.
Lenexa 66215
Tel: (913) 492-0500

KENTUCKY

Hamilton/Avnet Electronics
1051 D. Newton Park
Lexington 40511
Tel: (606) 259-1475

MARYLAND

Arrow Electronics, Inc.
8300 Guilford Drive
Suite H, River Center
Columbia 21046
Tel: (301) 995-0003
TWX: 710-238-9005

Hamilton/Avnet Electronics
6822 Oak Hall Lane
Columbia 21045
Tel: (301) 995-3500
TWX: 710-862-1861

†Mesa Technology Corp.
9720 Patuxent Woods Dr.
Columbia 21046
Tel: (301) 290-8150
TWX: 710-828-9702

†Pioneer/Technologies Group, Inc.
9100 Gaither Road
Gaithersburg 20877
Tel: (301) 921-0660
TWX: 710-828-0545

MASSACHUSETTS

Arrow Electronics, Inc.
25 Upton Dr.
Wilmington 01887
Tel: (617) 935-5134

†Hamilton/Avnet Electronics
10D Centennial Drive
Peabody 01960
Tel: (617) 531-7430
TWX: 710-393-0382

MTI Systems Sales
83 Cambridge St.
Burlington 01813

Pioneer Electronics
44 Hartwell Avenue
Lexington 02173
Tel: (617) 861-9200
TWX: 710-326-6617

MICHIGAN

Arrow Electronics, Inc.
755 Phoenix Drive
Ann Arbor 48104
Tel: (313) 971-8220
TWX: 810-223-6020

Hamilton/Avnet Electronics
2215 29th Street S.E.
Space A5
Grand Rapids 49508
Tel: (616) 243-8805
TWX: 810-274-6921

Pioneer Electronics
4504 Broadmoor S.E.
Grand Rapids 49508
FAX: 616-698-1831

†Hamilton/Avnet Electronics
32487 Schoolcraft Road
Livonia 48150
Tel: (313) 522-4700
TWX: 810-282-8775

†Pioneer/Michigan
13485 Stamford
Livonia 48150
Tel: (313) 525-1800
TWX: 810-242-3271

MINNESOTA

†Arrow Electronics, Inc.
5230 W. 73rd Street
Edina 55435
Tel: (612) 830-1800
TWX: 910-576-3125

†Hamilton/Avnet Electronics
12400 Whitewater Drive
Minnetonka 55434
Tel: (612) 932-0600

†Pioneer Electronics
7625 Golden Triange Dr.
Suite G
Eden Prairi 55343
Tel: (612) 944-3355

MISSOURI

†Arrow Electronics, Inc.
2380 Schuetz
St. Louis 63141
Tel: (314) 567-6888
TWX: 910-764-0882

†Hamilton/Avnet Electronics
13743 Shoreline Court
Earth City 63045
Tel: (314) 344-1200
TWX: 910-762-0684

NEW HAMPSHIRE

†Arrow Electronics, Inc.
3 Perimeter Road
Manchester 03103
Tel: (603) 668-6968
TWX: 710-220-1684

†Hamilton/Avnet Electronics
444 E. Industrial Drive
Manchester 03103
Tel: (603) 624-9400

NEW JERSEY

†Arrow Electronics, Inc.
Four East Stow Road
Unit 11
Marlton 08053
Tel: (609) 596-8000
TWX: 710-897-0829

†Arrow Electronics
6 Century Drive
Parsipanny 07054
Tel: (201) 538-0900

†Hamilton/Avnet Electronics
1 Keystone Ave., Bldg. 36
Cherry Hill 08003
Tel: (609) 424-0110
TWX: 710-940-0262

†Hamilton/Avnet Electronics
10 Industrial
Fairfield 07006
Tel: (201) 575-5300
TWX: 710-734-4388

†MTI Systems Sales
37 Kulick Rd.
Fairfield 07006
Tel: (201) 227-5552

†Pioneer Electronics
45 Route 46
Pinebrook 07058
Tel: (201) 575-3510
TWX: 710-734-4382

NEW MEXICO

Alliance Electronics Inc.
11030 Cochiti S.E.
Albuquerque 87123
Tel: (505) 292-3360
TWX: 910-989-1151

Hamilton/Avnet Electronics
2524 Baylor Drive S.E.
Albuquerque 87106
Tel: (505) 765-1500
TWX: 910-989-0614

NEW YORK

†Arrow Electronics, Inc.
3375 Brighton Henrietta Townline Rd.
Rochester 14623
Tel: (716) 275-0300
TWX: 510-253-4766

Arrow Electronics, Inc.
20 Oser Avenue
Hauppauge 11788
Tel: (516) 231-1000
TWX: 510-227-6623

Hamilton/Avnet
933 Motor Parkway
Hauppauge 11788
Tel: (516) 231-9800
TWX: 510-224-6166

†Hamilton/Avnet Electronics
333 Metro Park
Rochester 14623
Tel: (716) 475-9130
TWX: 510-253-5470

†Hamilton/Avnet Electronics
103 Twin Oaks Drive
Syracuse 13206
Tel: (315) 437-0288
TWX: 710-541-1560

†MTI Systems Sales
38 Harbor Park Drive
Port Washington 11050
Tel: (516) 621-6200

†Pioneer Electronics
68 Corporate Drive
Binghamton 13904
Tel: (607) 722-9300
TWX: 510-252-0893

Pioneer Electronics
40 Oser Avenue
Hauppauge 11787
Tel: (516) 231-9200

†Microcomputer System Technical Distributor Center

DOMESTIC DISTRIBUTORS (Cont'd.)

NEW YORK (Cont'd.)

†Pioneer Electronics
60 Crossway Park West
Woodbury, Long Island 11797
Tel: (516) 921-8700
TWX: 510-221-2184

†Pioneer Electronics
840 Fairport Park
Fairport 14450
Tel: (716) 381-7070
TWX: 510-253-7001

NORTH CAROLINA

†Arrow Electronics, Inc.
5240 Greensdairy Road
Raleigh 27604
Tel: (919) 876-3132
TWX: 510-928-1856

†Hamilton/Avnet Electronics
3510 Spring Forest Drive
Raleigh 27604
Tel: (919) 878-0819
TWX: 510-928-1836

Pioneer/Technologies Group, Inc.
9801 A-Southern Pine Blvd.
Charlotte 28210
Tel: (919) 527-8188
TWX: 810-621-0366

OHIO

Arrow Electronics, Inc.
7620 McEwen Road
Centerville 45459
Tel: (513) 435-5563
TWX: 810-459-1611

†Arrow Electronics, Inc.
6238 Cochran Road
Solon 44139
Tel: (216) 248-3990
TWX: 810-427-9409

†Hamilton/Avnet Electronics
954 Senate Drive
Dayton 45459
Tel: (513) 439-6733
TWX: 810-450-2531

Hamilton/Avnet Electronics
4588 Emery Industrial Pkwy.
Warrensville Heights 44128
Tel: (216) 349-5100
TWX: 810-427-9452

†Hamilton/Avnet Electronics
777 Brooksedge Blvd.
Westerville 43081
Tel: (614) 882-7004

†Pioneer Electronics
4433 Interpoint Boulevard
Dayton 45424
Tel: (513) 236-9900
TWX: 810-459-1622

†Pioneer Electronics
4800 E. 131st Street
Cleveland 44105
Tel: (216) 587-3600
TWX: 810-422-2211

OKLAHOMA

Arrow Electronics, Inc.
1211 E. 51st Street
Suite 101
Tulsa 74146
Tel: (918) 252-7537

†Hamilton/Avnet Electronics
12121 E. 51st St., Suite 102A
Tulsa 74146
Tel: (918) 252-7297

OREGON

†Almac Electronics Corp.
1885 N.W. 169th Place
Beaverton 97005
Tel: (503) 629-8090
TWX: 910-467-8746

†Hamilton/Avnet Electronics
6024 S.W. Jean Road
Bldg. C, Suite 10
Lake Oswego 97034
Tel: (503) 635-7848
TWX: 910-455-8179

Wyle Distribution Group
5250 N.E. Elam Young Parkway
Suite 600
Hillsboro 97124
Tel: (503) 640-6000
TWX: 910-460-2203

PENNSYLVANIA

Arrow Electronics, Inc.
650 Seco Road
Monroeville 15146
Tel: (412) 856-7000

Hamilton/Avnet Electronics
2800 Liberty Ave.
Pittsburgh 15238
Tel: (412) 281-4150

Pioneer Electronics
259 Kappa Drive
Pittsburgh 15238
Tel: (412) 782-2300
TWX: 710-795-3122

†Pioneer/Technologies Group, Inc.
Delaware Valley
261 Gibralter Road
Horsham 19044
Tel: (215) 674-4000
TWX: 510-665-6778

TEXAS

†Arrow Electronics, Inc.
3220 Commander Drive
Carrollton 75006
Tel: (214) 380-6464
TWX: 910-860-5377

†Arrow Electronics, Inc.
10899 Kinghurst
Suite 100
Houston 77099
Tel: (713) 530-4700
TWX: 910-880-4439

†Arrow Electronics, Inc.
2227 W. Braker Lane
Austin 78758
Tel: (512) 835-4180
TWX: 910-874-1348

†Hamilton/Avnet Electronics
1807 W. Braker Lane
Austin 78758
Tel: (512) 837-8911
TWX: 910-874-1319

TEXAS (Cont'd.)

†Hamilton/Avnet Electronics
2111 W. Walnut Hill Lane
Irving 75038
Tel: (214) 550-6111
TWX: 910-860-5929

†Hamilton/Avnet Electronics
4850 Wright Rd., Suite 190
Stafford 77477
Tel: (713) 240-7733
TWX: 910-881-5523

†Pioneer Electronics
18260 Kramer
Austin 78758
Tel: (512) 835-4000
TWX: 910-874-1323

†Pioneer Electronics
13710 Omega Road
Dallas 75234
Tel: (214) 386-7300
TWX: 910-850-5563

†Pioneer Electronics
5853 Point West Drive
Houston 77036
Tel: (713) 988-5555
TWX: 910-881-1606

Wyle Distribution Group
1810 Greenville Avenue
Richardson 75081
Tel: (214) 235-9953

UTAH

Arrow Electronics
1946 Parkway Blvd.
Salt Lake City 84119
Tel: (801) 973-6913

†Hamilton/Avnet Electronics
1585 West 2100 South
Salt Lake City 84119
Tel: (801) 972-2800
TWX: 910-925-4018

Wyle Distribution Group
1325 West 2200 South
Suite E
West Valley 84119
Tel: (801) 974-9953

WASHINGTON

†Almac Electronics Corp.
14360 S.E. Eastgate Way
Bellevue 98007
Tel: (206) 643-9992
TWX: 910-444-2067

Arrow Electronics, Inc.
19540 68th Ave. South
Kent 98032
Tel: (206) 575-4420

†Hamilton/Avnet Electronics
14212 N.E. 21st Street
Bellevue 98005
Tel: (206) 643-3950
TWX: 910-443-2469

Wyle Distribution Group
15385 N.E. 90th Street
Redmond 98052
Tel: (206) 881-1150

WISCONSIN

Arrow Electronics, Inc.
200 N. Patrick Blvd., Ste. 100
Brookfield 53005
Tel: (414) 767-6600
TWX: 910-262-1193

Hamilton/Avnet Electronics
2975 Moorland Road
New Berlin 53151
Tel: (414) 784-4510
TWX: 910-262-1182

CANADA

ALBERTA

Hamilton/Avnet Electronics
2816 21st Street N.E.
Calgary T2E 6Z3
Tel: (403) 230-3586
TWX: 03-827-642

Zentronics
Bay No. 1
3300 14th Avenue N.E.
Calgary T2A 6J4
Tel: (403) 272-1021

BRITISH COLUMBIA

†Hamilton/Avnet Electronics
105-2550 Boundary
Burmalay V5M 3Z3
Tel: (604) 437-6667

Zentronics
108-11400 Bridgeport Road
Richmond V6X 1T2
Tel: (604) 273-5575
TWX: 04-5077-89

MANITOBA

Zentronics
60-1313 Border Unit 60
Winnipeg R3H 0X4
Tel: (204) 694-1957

ONTARIO

Arrow Electronics, Inc.
36 Antares Dr.
Nepean K2E 7W5
Tel: (613) 226-6903

Arrow Electronics, Inc.
1093 Meyerside
Mississauga L5T 1M4
Tel: (416) 673-7769
TWX: 06-218213

†Hamilton/Avnet Electronics
6845 Rexwood Road
Units 3-4-5
Mississauga L4T 1R2
Tel: (416) 677-7432
TWX: 610-492-8867

Hamilton/Avnet Electronics
6845 Rexwood Road
Unit 6
Mississauga L4T 1R2
Tel: (416) 277-0484

ONTARIO (Cont'd.)

†Hamilton/Avnet Electronics
190 Colonnade Road South
Nepean K2E 7L5
Tel: (613) 226-1700
TWX: 05-349-71

†Zentronics
8 Tilbury Court
Brampton L6T 3T4
Tel: (416) 451-9600
TWX: 06-976-78

†Zentronics
155 Colonnade Road
Unit 17
Nepean K2E 7K1
Tel: (613) 226-8840

Zentronics
60-1313 Border St.
Winnipeg R3H 0l4
Tel: (204) 694-7957

QUEBEC

†Arrow Electronics Inc.
4050 Jean Talon Quest
Montreal H4P 1W1
Tel: (514) 735-5511
TWX: 05-25590

Arrow Electronics, Inc.
909 Charest Blvd.
Quebec J1N 2C9
Tel: (418) 687-4231
TWX: 05-13388

Hamilton/Avnet Electronics
2795 Halpern
St. Laurent H2E 7K1
Tel: (514) 335-1000
TWX: 610-421-3731

Zentronics
817 McCaffrey
St. Laurent H4T 1M3
Tel: (514) 737-9700
TWX: 05-827-535

intel

UNITED STATES
Intel Corporation
3065 Bowers Avenue
Santa Clara, CA 95051

JAPAN
Intel Japan K.K.
5-6 Tokodai, Tsukuba-shi
Ibaraki, 300-26

FRANCE
Intel Corporation S.A.R.L.
1, Rue Edison, BP 303
78054 Saint-Quentin-en-Yvelines Cedex

UNITED KINGDOM
Intel Corporation (U.K.) Ltd.
Pipers Way
Swindon
Wiltshire, England SN3 1RJ

WEST GERMANY
Intel Semiconductor GmbH
Dornacher Strasse 1
8016 Feldkirchen bei Muenchen

HONG KONG
Intel Semiconductor Ltd.
10/F East Tower
Bond Center
Queensway, Central

CANADA
Intel Semiconductor of Canada, Ltd.
190 Attwell Drive, Suite 500
Rexdale, Ontario M9W 6H8

Printed in U.S.A./C88-06/8K/0888/RRD RJ
Microprocessors

ISBN 1-55512-039-3