<< Manual Copy >> J. Eyles

Part IV, Chapter 3
IMAGE GENERATION CONTROLLER

IV.3.1 OVERVIEW

The Image Generation Controller (IGC) is a custom VLSI chip which supervises the bit-serial, SIMD execution of instructions on the 128 x 128 pixel processors contained on the 64 enhanced memory chips (EMC's) of a Renderer board. The IGC directly generates all signals for the EMC's, except for the Ph clock, the IO control signal (IOCtl), and the data nybble for the IO port. Additionally, the IGC controls other circuitry on the Renderer board (see Chapter III.4), such as that which causes transfers of data between the EMC's and the VRAM backing store, the Semaphore Counters, and the circuit which generates outgoing status messages from the IGC Port.

The IGC is implemented as a fully synchronous monolithic device using the HP CMOS40 1.6 micron CMOS technology. It is packaged in the MOSIS 132-pin grid array standard package (14x14 pin arrangement, 0.45" die cavity). It sports approximately 126,000 transistors.

The IGC accepts commands consisting of an opcode, optionally accompanied by a supplementary opcode and/or coefficients for the quadratic expression evaluators (QEE's) of the EMC's, and generates the cycle-by-cycle instructions required to execute the instruction in the pixel processors of the EMC's.

The IGC commands are described in detail in the Renderer documentation (Chapter III.4); they are roughly similar to those described in the "Pxpl4 Programmers Guide". Notably absent in Pxpl5 are the commands FB_VRWAIT, and FB_FRWAIT, since the IGC does not communicate directly with the video system, and FB_REFRESH, since Pixel-Planes 5 has static pixel-memory. New commands are included: to handle transfers of data between the EMC's and the VRAM backing store; to configure the QEE's of the EMC's; to cause a status message to be generated; and a set of commands for sending integer data directly to the pixel-ALU's, bypassing the QEE.

The IGC is based on a microcode sequencer, which generates the ALU micro-instruction and memory read/write signals for the EMC's directly, along with control signals for the remainder of the IGC; the Sequencer uses two loop counters, which are initialized from fields in the instruction opcode. Pixel-memory addresses for the EMC's are generated by a set of 3 address counters, which are initialized from fields in the instruction op code and controlled by the microcode sequencer. The 6 coefficient bit-streams for the QEE's are generated by a set of shift registers, which convert the parallel IEEE standard floating-point format supplied with a command into a bit-serial 2's-complement fixed-point representation with 0 to 30 fractional bits of precision. These coefficient shifters also are controlled by the microcode sequencer. Another shifter allows an integer coefficient to be shifted directly into the pixel-ALU's of the EMC's.

In order to achieve optimum performance, two complications must be introduced into this scheme: first, the magnitude of the coefficients, and hence the number of microcycles necessary to form the full QEE result, will vary from instruction to instruction, so the length of an instruction is not known *a priori*; second, it is desirable to overlap tree operations — shifting the coefficients for one instruction into the QEE's, while the previous instruction is still being executed at the pixel-ALU's and pixel-memory. The solution to these problems is to provide, in addition to shift registers to bit serialize the coefficients, a mechanism which provides signals marking the location of the LSB and MSB of the coefficients and QEE results. These LSB and MSB signals can be used as condition flags for the microcode sequencer, thereby synchronizing the pixel-ALU and pixel-memory micro-instructions with the QEE's; this allows bit serialization (and shifting into the QEE's) of the coefficients of an instruction to begin as soon as the correct number of bits of the coefficients of the previous instruction have been generated. Thus coefficients from the next instruction may be bit-serialized even while execution of the current instruction is being completed, thereby allowing overlapping.

Instructions pass through what is essentially a 3-stage pipeline in the IGC. In the first stage, an input device writes the instruction to 8 virtual input registers in the IGC. In the second stage, the fields in the opcode are transferred to another set of latches and bit-serialization of the A,B,C,D,E,F coefficients begins (so that the coefficient bit-streams can begin being shifted into the QEE's of the EMC's). In the third stage, execution of the micocode for the instruction takes place. An instruction enters the third stage of the pipe when execution of microcode for the previous instruction has completed. Serialization of the coefficients continues if necessary, the loop count and pixel-memory address fields from the opcode are loaded into their respective counters, and the microcode sequencer begins executing microcode at the starting address specified in the opcode.

Pinout for the device is shown in Figure IV.3-1.

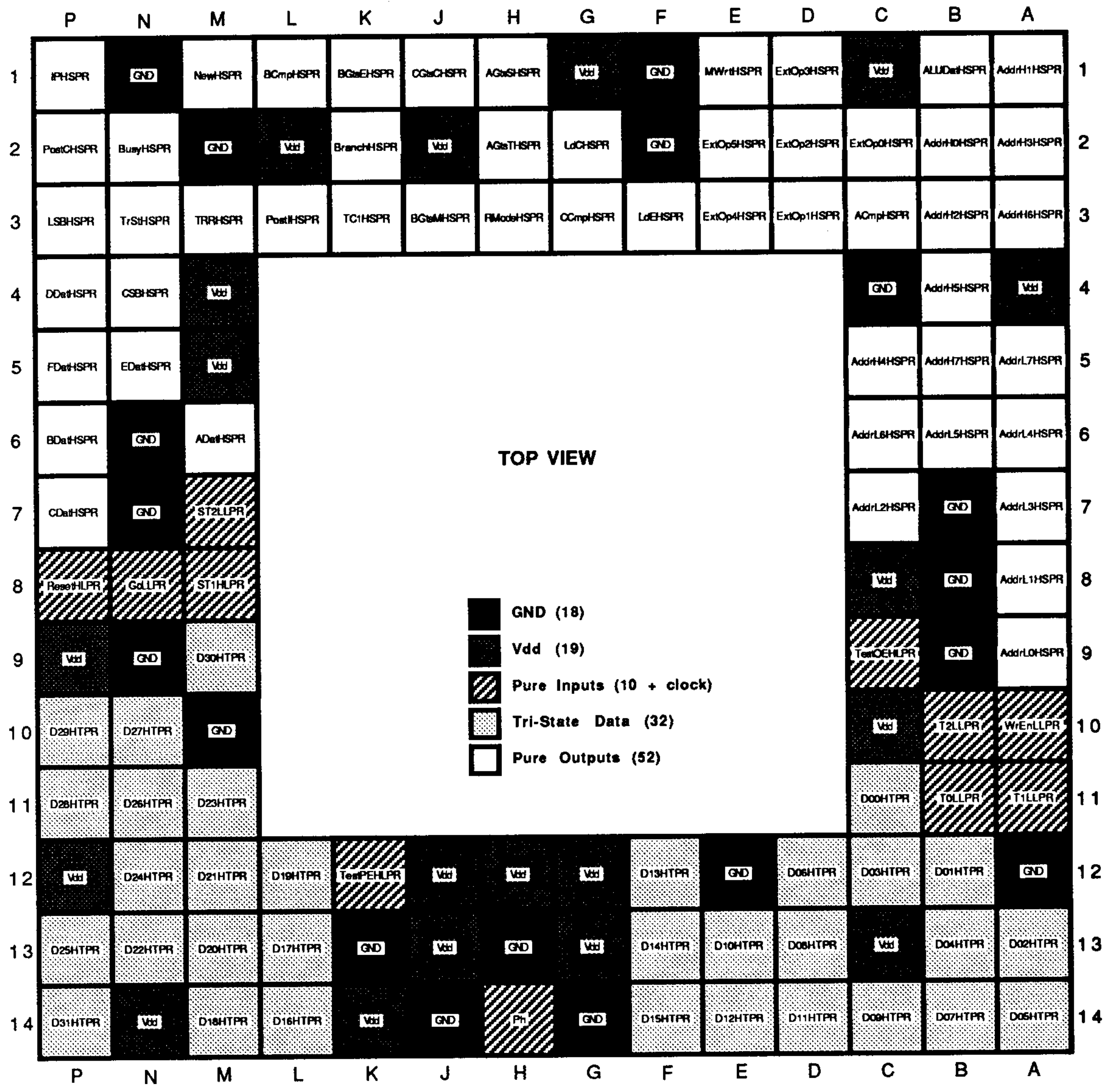

P N M L K J H G F E D C B A
1 IPHSPR GND NewHSPR BCmpHSPR BGtsEHSPR CGtsCHSPR AGtsSHSPR Vdd GND MWrtHSPR ExtOp3HSPR Vdd ALUDatHSPR AddrH1HSPR 1
2 PostCHSPR BusyHSPR GND Vdd BranchHSPR Vdd AGtsTHSPR LdCHSPR GND ExtOp5HSPR ExtOp2HSPR ExtOp0HSPR AddrH0HSPR AddrH3HSPR 2
3 LSBHSPR TrStHSPR TRRHSPR PostIHSPR TC1HSPR BGtsMHSPR RModeHSPR CCmpHSPR LdEHSPR ExtOp4HSPR ExtOp1HSPR ACmpHSPR AddrH2HSPR AddrH6HSPR 3
4 DDatHSPR CSBHSPR Vdd GND AddrH5HSPR Vdd 4
5 FDatHSPR EDatHSPR Vdd AddrH4HSPR AddrH7HSPR AddrL7HSPR 5
6 BDatHSPR GND ADatHSPR AddrL6HSPR AddrL5HSPR AddrL4HSPR 6
7 CDatHSPR GND ST2LLPR AddrL2HSPR GND AddrL3HSPR 7
8 ResetHLPR GoLLPR ST1HLPR Vdd GND AddrL1HSPR 8
9 Vdd GND D30HTPR TestOEHLPR GND AddrL0HSPR 9
10 D29HTPR D27HTPR GND Vdd T2LLPR WrEnLLPR 10
11 D28HTPR D26HTPR D23HTPR D00HTPR T0LLPR T1LLPR 11
12 Vdd D24HTPR D21HTPR D19HTPR TestPEHLPR Vdd Vdd Vdd D13HTPR GND D06HTPR D03HTPR D01HTPR GND 12
13 D25HTPR D22HTPR D20HTPR D17HTPR GND Vdd GND Vdd D14HTPR D10HTPR D08HTPR Vdd D04HTPR D02HTPR 13
14 D31HTPR Vdd D18HTPR D16HTPR Vdd GND Ph GND D15HTPR D12HTPR D11HTPR D09HTPR D07HTPR D05HTPR 14
P N M L K J H G F E D C B A
TOP VIEW
GND (18)
Vdd (19)
Pure Inputs (10 + clock)
Tri-State Data (32)
Pure Outputs (52)

Figure IV.3 -1: IGC 5 Pin Out

(Package is Kyocera KD-83560-A 132-pin PGA.)

IV.3.2 FUNCTIONAL SPECIFICATION

IV.3.2—1 External Interface Signals

Signal Name	Pin	Description
		Power/Ground
Vdd (clocks)	K14 J12 J13 H12 G12 G13	Nominally +5.0 volts, the power supply pins are in 3 groups. These power the two-phase clock generator.
Vdd (internal)	G1 L2 M5 C8 P9 P12 N14 C13 C10	Power pins for the input and IO pads, and the chip internals, including the memory clock generators.
Vdd (outputs)	C1 J2 M4 A4	Power pins for the output pads. The power system for the output pads is separate from the main power system, but shorted to it by an internal resistance of approximately 10 ohms.
GND (clocks)	K13 J14 H13 G14	The ground pins are also divided into 3 groups. These are the ground pins for the two-phase clock generators.
GND (internal)	F2 M2 B7 N7 N9 M10 E12 A12 B9	Ground pins for the input and IO pads, and the chip internals, including the memory clock generators.
GND (outputs)	F1 N1 N6 B8 C4	Ground pins for the output pads. The ground system for the output pads is separate from the main ground system, but shorted to it by a resistance of approximately 10 ohms.

Input and Input/Output

Ph	H14	The IGC system clock. All I/O are referred to the rising edge of this clock, whose nominal frequency is 40.0 MHz. Duty factor for this signal should be closely controlled to 50%.
ResetHLPR	P8	Reset signal, should be asserted for one or more cycles, with WrEnLLPR and GoLLPR held high, to initialize the chip.
TestPEHLPR	K12	These signals are used to read the microcode memory during chip testing (see Section IV.3.2—X below). They must be grounded during normal operation.
TestOEHLPR	C9	
D00HTPR	C11	The data pins, used to write command words into the IGC input registers. During chip testing they are used as IO pins to read microcode memory, but are purely inputs in normal operation.
D01HTPR	B12	
D02HTPR	A13	
D03HTPR	C12	
D04HTPR	B13	
D05HTPR	A14	
D06HTPR	D12	
D07HTPR	B14	
D08HTPR	D13	
D09HTPR	C14	
D10HTPR	E13	
D11HTPR	D14	
D12HTPR	E14	
D13HTPR	F12	
D14HTPR	F13	
D15HTPR	F14	
D16HTPR	L14	
D17HTPR	L13	
D18HTPR	M14	
D19HTPR	L12	
D20HTPR	M13	
D21HTPR	M12	
D22HTPR	N13	
D23HTPR	M11	
D24HTPR	N12	
D25HTPR	P13	
D26HTPR	N11	
D27HTPR	N10	
D28HTPR	P11	
D29HTPR	P10	
D30HTPR	M9	
D31HTPR	P14	
T0LLPR	B11	Active low register address pins (T0LLPR is LSB), select which of the 8 input registers is to be written when WrEnLLPR is asserted.
T1LLPR	A11	
T2LLPR	B10	

WrEnLLPR	A10	On a cycle when this signal is low, the data on D<0:31>HTPR is written to the input register selected by T<0:2>LLPR. WrEnLLPR must not be asserted on successive cycles, or when the output BusyHSPR is asserted.
GoLLPR	N8	Active low, this signal is asserted for a single cycle after the last word of an instruction has been written. It may be asserted on the same cycle as the last input register is loaded, or on a subsequent cycle. It must not be asserted on successive cycles, or when the output BusyHSPR is asserted.
ST1HLPR **ST2LLPR**	M8 M7	Condition inputs, used to qualify conditional branches during microcode sequencing. ST1HLPR causes branch when low, ST2LLPR causes branch when high.

Outputs

BusyHSPR	N2	Handshaking signal for IO interface. Goes high after GoLLPR is asserted. Neither WrEnLLPR nor GoLLPR may be asserted until BusyHSPR goes low again.
TrStHSPR	N3	These are the control signals for the EMC array. Normally each of these signals would drive the corresponding signal on each EMC on a Renderer (i.e.: TrStHSPR would drive all TrStHLPR's). See EMC spec for detailed descriptions of their functions. They may also be used in conjunction with ExtOp<0:5>HSPR to control other devices on the Renderer.
LSBHSPR	P3	
ADatHSPR	M6	
BDatHSPR	P6	
CDatHSPR	P7	
DDatHSPR	P4	
EDatHSPR	N5	
FDatHSPR	P5	
AGtsTHSPR	H2	
AGtsSHSPR	H1	
ACmpHSPR	C3	
BGtsMHSPR	J3	
BGtsEHSPR	K1	
BCmpHSPR	L1	
CGtsCHSPR	J1	
CCmpHSPR	G3	
LdEHSPR	F3	
LdCHSPR	G2	
MWrtHSPR	E1	

AddrL0HSPR	A9	These signals drive the address inputs, Add<0:7>HLPR, to the EMC's. Normally AddrL*i*HSPR and AddrH*i*HSPR are identical, but under certain conditions they can be made a complementary pair. Normally the EMC's on a Renderer would be connected to these signals in such a way that EMC number *j* is connected to AddrL*i*HSPR if bit *i* of *j* is 0, to AddrH*i*HSPR if bit *i* of *j* is 1. This allows the EMC's to be addressed separately using the special mode.
AddrL1HSPR	A8	
AddrL2HSPR	C7	
AddrL3HSPR	A7	
AddrL4HSPR	A6	
AddrL5HSPR	B6	
AddrL6HSPR	C6	
AddrL7HSPR	A5	
AddrH0HSPR	B2	They may also be used in conjunction with ExtOp<0:5>HSPR to control other devices on the Renderer.
AddrH1HSPR	A1	
AddrH2HSPR	B3	
AddrH3HSPR	A2	
AddrH4HSPR	C5	
AddrH5HSPR	B4	
AddrH6HSPR	A3	
AddrH7HSPR	B5	
ALUDatHSPR	B1	The output of the shifter for the integer C coefficient.
ExtOp0HSPR	C2	Strobes used to control other devices on the Renderer (other than the EMC's). No more than one of ExtOp<0:5>HSPR can be asserted on a given cycle.
ExtOp1HSPR	D3	
ExtOp2HSPR	D2	
ExtOp3HSPR	D1	
ExtOp4HSPR	E3	
ExtOp5HSPR	E2	
BranchHSPR	K2	These are diagnostic signals used when testing the IGC. The represent the similarly named on-chip signal delayed by one or two clock cycles. In normal use they should be "no connects".
NewHSPR	M1	
TC1HSPR	K3	
PostCHSPR	P2	
PostIHSPR	L3	
IPHSPR	P1	
RModeHSPR	H3	
TRRHSPR	M3	
CSBHSPR	N4	

IV.3.2—2 External Interface Specifications

Operating Conditions

Parameter		Min	Nom	Max	Units
V_{dd}	Power supply voltage	4.5	5.0	5.5	Volts
V_{SS}	Power supply return		0		Volts
V_{IH}	Logic-HI input voltage	2.0		5.5	Volts
V_{IL}	Logic-LO Input voltage	-1.2		0.8	Volts
T_A	Ambient air temperature	0	25	40	C
T_J	Junction temperature	25	50	70	C

DC Electrical Characteristics (V_{dd} = 5.0 volts)

Parameter		Conditions	Min	Nom	Max	Units
V_{OH}	HI output voltage, TPR,SPR outputs	I_{OH} = -20ma	2.4			Volts
V_{OL}	LO output voltage, TPR,SPR outputs	I_{OL} = 20ma			0.36	Volts
I_{ISPR}	Input leakage current, LPR inputs	$0 \leq V_{in} \leq V_{dd}$			10	µA
I_{BH}	Input clamp current, I/O's HI	Vin = Vdd + 1 volt			70	mA
I_{BL}	Input clamp current, I/O's LO	Vin = - 1 volt			-70	mA
P_D	Power dissipation, f_{Ph} = 40.0 MHz					W

Capacitance[4]

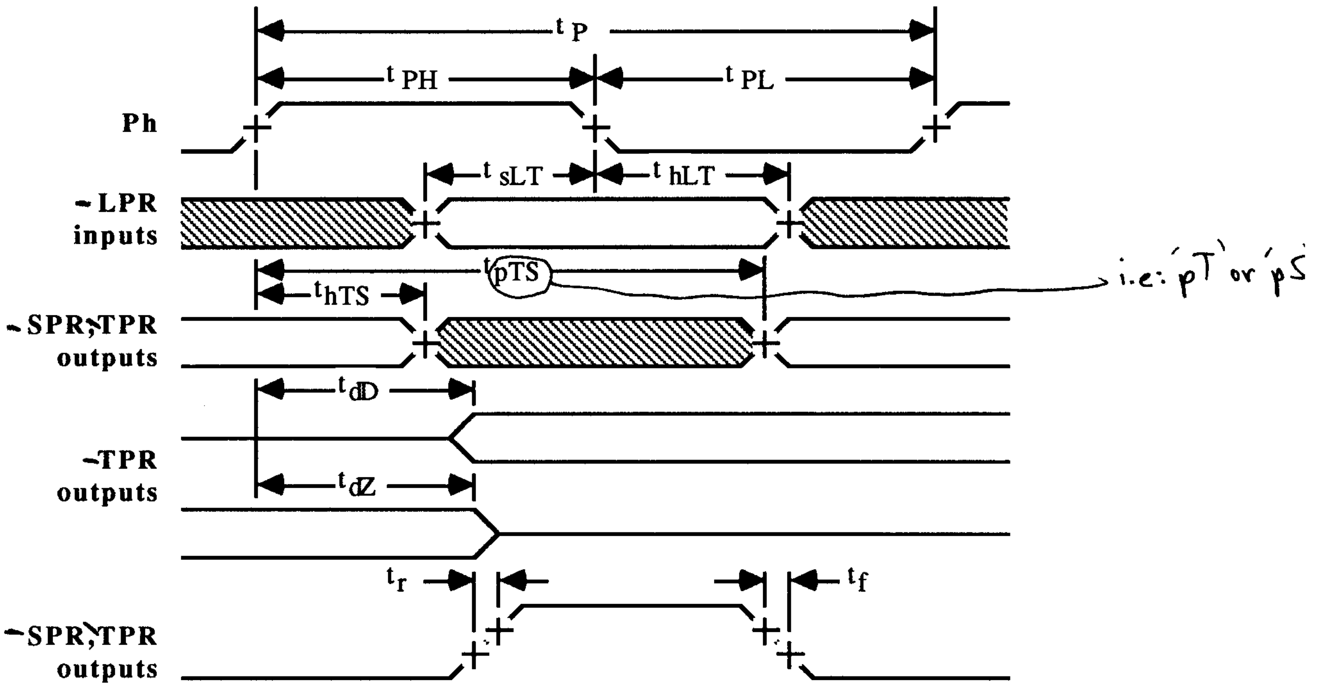
Parameter		Min	Nom	Max	Units
C_{LPR}	Input capacitance, LPR inputs		3.7		pF
C_{TPR}	Capacitance, TPR I/O pins		3.7		pF
C_{Ph}	Input capacitance, Ph clock input		5.4		pF
C_{SPR}	~~[illegible]~~ Capacitance SPR outputs				pF

Timing Requirements (Timing measurements are referred to Vin = 1.40 volts.)

Parameter		Min	Nom	Max	Units
t_P	Clock period		25		ns
t_{PH}	Clock HI period		12.5		ns
t_{PL}	Clock LO period		12.5		ns
t_{sLT}	Setup time, LPR,TPR inputs	0.0			ns[1]
t_{hLT}	Hold time, LPR,TPR inputs	9.0			ns[1]

Timing Characteristics

Parameter		Conditions	Min	Nom	Max	Units
tdD	Delay, clock edge to data drive	CL=50Ω‖50pf	5.4		22	ns1
tdZ	Delay, clock edge to data Hi-Z	CL=50Ω‖50pf	5.4		23	ns1
trTS	Rise time, TPR, SPR outputs	CL=50Ω‖50pf			5.6	ns1
tfTS	Fall time, TPR, SPR outputs	CL=50Ω‖50pf			4.1	ns1
thT	Guaranteed hold time, TPR outputs	No Load	9			ns1
tpT	Worst-case propagation time, TPR outputs	CL=50Ω‖50pf	25			ns1
thS	Guaranteed hold time, SPR output	No Load	9			ns1
tpS	Worst-case propagation time, SPR output	CL=50Ω‖50pf	21			ns1



Notes:

1 Based on CAzM simulation. Max. times: Worst conditions; Min. times: Best conditions.

IV.3.2—3 Timing of External Interface Signals

The input signals bear the signal name suffix **-LPR**; this indicates that the inputs should be of type "Latched on **Ph** **R**ising edge"; specifically, these signals must be stable in a window about the falling edge of **Ph**. It is intended that the inputs be driven by an edge-triggered latch clocked on the rising edge of **Ph**.

The IO signals, **D<0:31>HTPR**, behave just like **-LPR** input signals except in test mode (see below).

The output signals bear the signal name suffix **-SPR**; this indicates that the outputs are guaranteed to be stable in a window about the rising edge of **Ph**. It is intended that they be latched off-chip by an edge-triggered latch clocked on the rising edge of **Ph**.

IV.3.2 — 3 Input Protocol

The IGC accepts command input. Commands are written by loading one of more of the eight 32-bit input registers and then asserting **GoLLPR**. To load a register, **WrEnLLPR** is asserted for one clock cycle; on the same clock cycle, **T<0:2>LLPR** specify the desired register, and **D<0:31>HTPR** contain the data to be loaded. **WrEnLLPR** should never be asserted on successive clock cycles, nor should it be asserted when **BusyHSPR** is high. The 8 input registers of the IGC are defined as follows:

T2	T1	T0	MNEMONIC	INTERPRETATION OF D<0:31>HTPR
0	0	0	I	instruction opcode (see Fig. IV.3.1)
0	0	1	P	supplementary opcode (see Fig. IV.3.2)
0	1	0	A	A coefficient, as IEEE single-precision float
0	1	1	B	B coefficient, as IEEE single-precision float
1	0	0	C	C coefficient, as IEEE float or 32-bit integer
1	0	1	D	D coefficient, as IEEE single-precision float
1	1	0	E	E coefficient, as IEEE single-precision float
1	1	1	F	F coefficient, as IEEE single-precision float

Note: in this table, the sense of T0-T2 is "active high"; the actual chip inputs are active low

These IGC *registers* are virtual registers. In fact, a write to one of the 8 registers may result in different portions of the data word being loaded into different IGC modules, and some portions written into several modules.

The value written into any virtual register overwrites the previously written value. The previously written value for a register can be re-used, simply by not writing a new value; however, if IGC microcode is reloaded, or if FBITS is altered (see below), old register contents are no longer valid.

Once all the appropriate registers for a command have been loaded, **GoLLPR** is asserted for one cycle, indicating that the command is loaded into the input registers and is ready to be processed. **GoLLPR** may be asserted with **WrEnLLPR** when the last word of the instruction is written, or on a subsequent cycle.

When **GoLLPR** is asserted to conclude the transmission of an instruction, **BusyHSPR** goes high on the next cycle, indicating that the IGC is "busy", and no further writes to the IGC are permitted. Neither **WrEnLLPR** nor **GoLLPR** should be asserted again, until **BusyH** goes low. When **BusyH** goes low, a new instruction may be loaded; while the new instruction is written, **BusyH** remains low until **Go** is asserted again

IV.3.2—4 Input Data Formats

I Register. The format of the I register data (the *opcode*) is shown in Fig. IV.3-2.

Bits 8-16 (the *MCAddr* field) define the starting address in the microcode sequencer for execution of the instruction.

Bits 0-7 (the *DstAddr* field) define the *destination* pixel-memory operand, by its LSB.

Bits 18-19 (the *QEEMode* field) specify the function of the quadratic expression evaluator (QEE) for the instruction. If QEEMode is zero, the instruction does *not* use the QEE; if the field is non-zero then the instruction uses the QEE, in one of three modes:

Bit 19	Bit 18	Mode	QEE function
0	0	——	QEE result not used
0	1	constant	$Q(x,y) = C$
1	0	linear	$Q(x,y) = Ax + By + C$
1	1	quadratic	$Q(x,y) = Dx^2 + Exy + Fy^2 + Ax + By + C$

In *quadratic* mode, the full bi-quadratic expression is computed in the QEE. If *linear* mode is specified, the QEE computes just the linear portion, Ax + By + C, and the D, E, and F coefficients are ignored; the D,E, and F registers need not be explicitly zero-ed, and the previously written D,E, and F values will still be in the registers if it is desired to re-use previously written D,E,F values when running the QEE in quadratic mode in a subsequent instruction. Similarly, *constant* mode causes the QEE to just compute the constant C, without requiring 0's to be written to the other coefficients and without disturbing the contents of those registers.

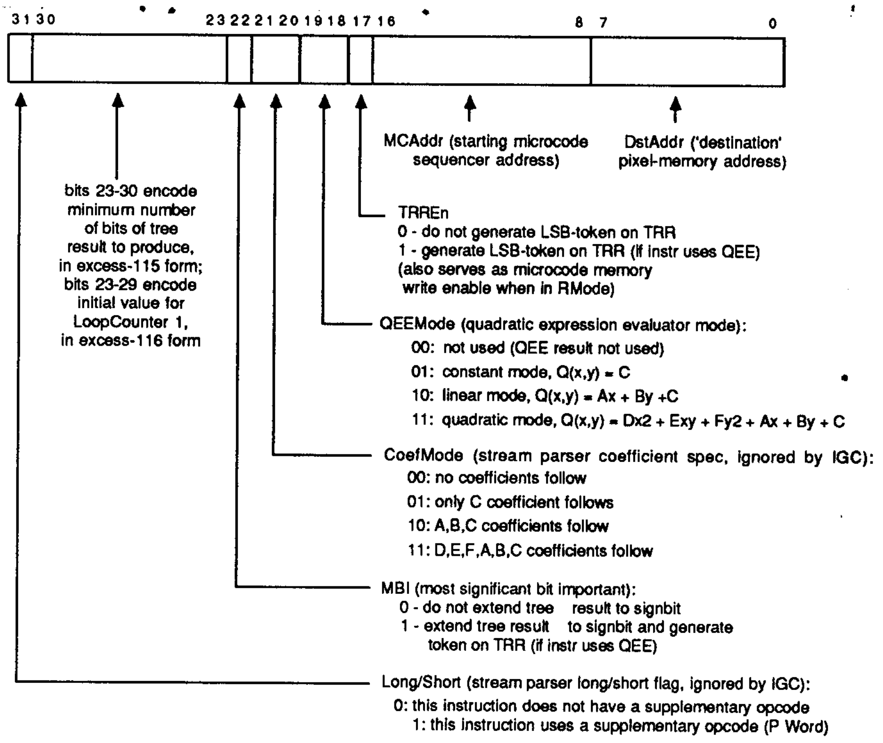
Bits 23-30 (the *LpCntl/FNI* field) have a dual function. Bits 23-29 specify the starting value for Loop Counter 1 (LpCnt1), in excess-116 form. If the instruction uses the QEE result (bits 18 and 19 are not both zero), bits 23-30 also specify the minimum number of integer bits of the QEE result to be generated (FNI), in excess-115 form. So for instructions which use the QEE result, the starting count for Loop Counter 1 is fixed and must be equal to FNI - 1.

Bit 17 (the *TRREn/WE* bit) indicates that a token is to be generated, on the **Serializer** output **TRRHSP1**, marking the LSB of the QEE result. It also serves as a microcode memory write-enable, when the IGC is in RMode.

Bit 22 (the *MBI* bit) specifies that the coefficient bit-streams must be extended to sufficient length to insure that the sign bit of the QEE result is generated at all pixel processors, and that a token, on the **Serializer** output **TRRHSP1**, marking the sign-bit of the QEE result.

Bits 20, 21, and 31 are ignored by the IGC itself. They are reserved for use by the Stream Parser on the Renderer board, to specify which registers are to be loaded for a specific instance of an instruction. These 3 bits, along with bit 19, the MSB of the QEEMode field, compose a 4-bit stream parser code; this is described fully in the Renderer documentation (Section III.4).

Format for I Word


31 30
23 22 21 20 19 18 17 16
8 7
0
bits 23-30 encode minimum number of bits of tree result to produce, in excess-115 form; bits 23-29 encode initial value for LoopCounter 1, in excess-116 form
MCAddr (starting microcode sequencer address)
DstAddr ('destination' pixel-memory address)
TRREn
0 - do not generate LSB-token on TRR
1 - generate LSB-token on TRR (if instr uses QEE) (also serves as microcode memory write enable when in RMode)
QEEMode (quadratic expression evaluator mode):
00: not used (QEE result not used)
01: constant mode, Q(x,y) = C
10: linear mode, Q(x,y) = Ax + By +C
11: quadratic mode, Q(x,y) = Dx2 + Exy + Fy2 + Ax + By + C
CoefMode (stream parser coefficient spec, ignored by IGC):
00: no coefficients follow
01: only C coefficient follows
10: A,B,C coefficients follow
11: D,E,F,A,B,C coefficients follow
MBI (most significant bit important):
0 - do not extend tree result to signbit
1 - extend tree result to signbit and generate token on TRR (if instr uses QEE)
Long/Short (stream parser long/short flag, ignored by IGC):
0: this instruction does not have a supplementary opcode
1: this instruction uses a supplementary opcode (P Word)

Format for P Word

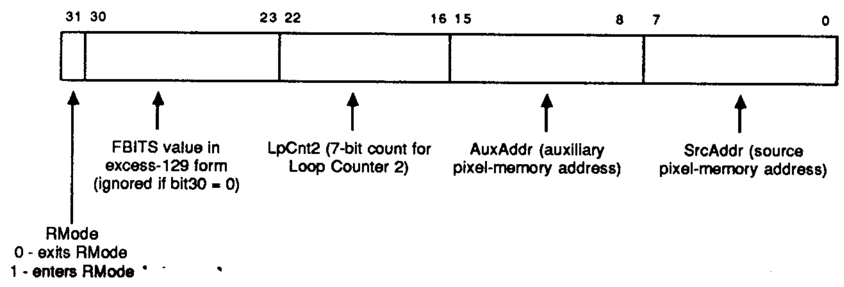

31 30
23 22
16 15
8 7
0
FBITS value in excess-129 form (ignored if bit30 = 0)
LpCnt2 (7-bit count for Loop Counter 2)
AuxAddr (auxiliary pixel-memory address)
SrcAddr (source pixel-memory address)
RMode
0 - exits RMode
1 - enters RMode

Figure IV.3 - 2 : I and P Opcodes

Supplementary Opcode. The format of the P register data (the *supplementary opcode*) is shown in Fig. IV.4.2.2.

Bits 0-7 (the *SrcAddr* field), specify the *source* pixel-memory operand, by its LSB.

Bits 8-15 (the *AuxAddr* field), specify the *auxiliary* pixel-memory operand, by its LSB.

Bits 16-22 (the *LpCnt2* field) specify the starting value for Loop Counter 2.

Bits 23-30 (the *FBITS* field) are used for setting FBITS, the number of fractional bits generated when the IGC converts the IEEE floating-point coefficients supplied with an instruction into fixed-point numbers. FBITS may be bewteen 0 and 30, and is encoded in bits 23-30 in excess-129 form. Bits 23-30 must either be in the legal range 129 to 159, or else they must be zero. After changing the FBITS setting, all QEE coefficients registers must be reloaded, since previously loaded values are no longer valid. FBITS should not be altered as part of an instruction which uses the QEE's.

Bit 31 (the *RMode* bit) is used to enter and exit the special initialization mode RMode. When the P register is written with bit 31 = 1, the IGC enters RMode immediately (without **GoLLPR** being asserted), and when the P register is written with bit 31 = 0, the IGC leaves RMode immediately.

In creating P register data, care must be taken that bits 30 and 31 be 0, unless it is desired to enter RMode or to change the FBITS setting.

A,B,C,D,E,F Registers. The formats for the A,B,C,D,E, and F words are just the IEEE standard single-precision floating-point format: bits 0-22 represent the fractional portion of the mantissa, with an understood 1 to the left of the radix point; bits 23-30 represent the exponent in excess-127 form; bit 31 is the sign-bit (the representation is sign/magnitude). The IEEE standard specifies several exceptions, with exponent fields of all 0's or all 1's; the IGC does not recognize these exceptions as such. The exponent for an exception is always outside of the valid range for exponents, which is -FBITS through 63-FBITS; any input coefficient with an exponent outside this range is treated as zero.

The C register is also used for direct-to-ALU mode and EMC configuration. It this mode, the C value is treated as a 32-bit signed integer and is shifted into the ACmp input of the EMC's. Integer data may not be written to the C register for use by the QEE's.

IV.3.2 — 5 Generating Command Input for the IGC

As described above, commands are written to the IGC by writing: the opcode to the I register; the supplementary opcode, if required, to the P register; quadratic coefficents, if required, to the A, B, C, D, E, and F registers; and scalar, if required, to the C register.

The opcode and supplementary opcodes are generated using macros produced by the microcode assembler *asmpp5* in the header file *igc_opcodes.h*. For a given command NEM (args), the macro I_NEM generates most of the opcode from the specified 'args'. However, the user may need to modify the QEEMode and Stream Parser coefficient specification fields. If the QEEMode bits in the macro generated opcode are 00, the instruction does not use the QEE's and the bits must not be modified. However, if the instruction does use the QEE's, the macro generated opcode will have the QEEMode bits set to 11. If the user wishes to run the QEE in constant or linear mode, this field must be altered as shown in the table above. Setting of the Stream Parser coefficient specification field is described in the Renderer documentation. The supplementary opcode is generated by the macro P_NEM. No user modification of this automatically generated opcode is required or allowed.

The coefficient data words are the IEEE standard single-precision floating-point format.

Valid values for the coefficients are with exponents in the range -FBITS to 63-FBITS. Coefficients with magnitudes outside this range are treated as 0; this includes very large and very small numbers, zero, and the exceptions defined in the IEEE standard. Because the coefficients are converted to fixed-point numbers, precision is often lost . The precision lost is determined by the magnitude of the coefficient and the setting for the "number of fractional bits" (FBITS). The floating-point value is *truncated*, not *rounded*, when the conversion is made, and the truncation is towards zero. Further, although FBITS fractional bit of the coefficients are generated, the corresponding FBITS fractional bits of the QEE results are not available. Only the integer portion of the QEE results are available at the pixel-ALU and pixel-memory. Thus, two truncations occur when floating-point coefficients are supplied to the EMC's, which are essentially integer devices: (1) the coefficients are truncated to fixed-point, (2) the QEE result is truncated downwards (not towards zero) to an integer.

IV.3.2—6 Changing Number of Fractional Bits

The IGC converts the IEEE standard floating-point coefficients written to the A,B,C,D,E, and F registers with a command, to a bit-serial 2's complement fixed-point representation required by the QEE's. The number of fractional bits in the fixed-point representation (FBITS) is variable between 0 and 30.To change FBITS, a word is written to the P register, with bits 23-30 (the exponent field) encoding the desired value as FBITS + 129. Otherwise, bit 30 must be 0 whenever the P register is written and it is not desired to alter FBITS.

Normally, a separate command is used to change FBITS. However, if an algorithm requires frequent changes of FBITS, IGC input traffic can be reduced by changing FBITS as part of a regular command. If this is done, the P register must be written either before or after all of the other words of the command, else their exponent fields will be treated differently. Since the Stream Parser always writes the P word after the I opcode and before the coefficients, FBITS can only be changed as part of a command which does not use the QEE.

IV.3.2—7 Initialization and Loading the Microcode Store

On power-up, if the microcode is faulty, if invalid opcodes are given, or if the IGC gets stuck waiting on an external handshake signal, the IGC is likely to be *hung*, that is, **BusyHSPR** is stuck high and the inputting device cannot write to the IGC. To unstick a hung IGC, to load microcode store at initialization time, or to swap out microcode on-the-fly, the IGC must be placed into a special mode called RMode.

Entering and exiting RMode. RMode can be entered in 2 ways: (1) by asserting the input **ResetHLPR** with **GoLLPR** and **WrEnLLPR** = 1, or (2) by writing to the P register with D31 = 1. If the IGC is hung, **ResetHLPR** can be used to force RMode, or the inputting device may ignore **BusyHSPR** and use a P register write to enter RMode. RMode is exited by writing to the P register with D31 = 0. Any write to the P register must insure that bit 31 is set or cleared acording to whether it is desired to stay in or out of RMode. A write to the P register will cause the IGC to enter or exit RMode immediately; **GoLLPR** need not be asserted.

Effects of RMode. RMode has the following effects on the IGC: **ShiftHSP2** is always asserted; the **CSBHSP1** signal from the **Sequencer** is ignored by the **Controller** so that it cannot inhibit **PostCHSP2**; the **Sequencer** always asserts **NewLSP1**, so the "new" microcode address is always used; **IPHSP1** always causes **MWriteHSP1**, if the write enable bit of the I word is set; and the IGC outputs **LdEHSPR**, **LdCHSPR**, **MWrtHSPR**, **CCmpHSPR**, and **ExtOp<0:5>HSPR** are forced to 0, to avoid corrupting the EMC state or triggering logic on the Renderer board during an on-the-fly microcode reload.

Microcode loading. To load or reload the microcode store, the IGC is first put into RMode. If **ResetHLPR** is used to enter RMode, it must be de-asserted before proceeding.

If **BusyHSPR** is ignored to enter RMode, the normal input protocol must be obeyed for all subsequent input.

If microcode is loaded "on-the-fly", RMode can be entered just by writing to the P register with bit 31 = 1. This must be proceeded by two no-op instructions (writing 0's to the I register) to insure that the instructions immediately prior to the microcode reload are not corrupted.

For each microcode location to be loaded, a 2-word command is sent. The opcode, written to the I register, contains the memory address to be loaded in bits 8-16, bit 17 is set to enable the write, and the stream parser coefficient mode bits are set for "C only" mode, that is, bit 18 is set; all the other bits should be 0. Next, the microcode word to be loaded is written to the C register. Finally, **GoLLPR** is asserted, as with a normal command.

no, bit 20!

After all desired microcode locations are loaded, a one-word command is sent with all 0's opcode. This sets the **Sequencer** to address 0, the idle state.

Finally, RMode is exited, by writing to the P register with bit 31 = 0, and normal operation begins.

It is important that (1) all opcodes written during this loading process have the QEE Mode bits and the MBI bit set to 0, and (2) the initialization sequence last at least 100 or so clock cycles after the first such instruction; this insures that the "coefficient serializers busy" logic and the various shifters in the **Serializer** are flushed. The bits of the P Word, other than bit 31, do not matter, except that if bit 30 is 1 the FBITS register will be altered. If desired, FBITS may be set in the normal way while in RMode, as part of the initialization sequence, for example, as part of the command which exits RMode.

The Renderer document (Chapter III.4) describes commands for performing the initialization sequence.

Reading the Microcode store. While in RMode, the microcode store may be read as well as written. First, a one word (opcode only) command is sent, with bits 8-16 containing the address to be read, and all other bits 0. Next, the IGC data port is put in output mode, by first simultaneously tri-stating **D<0:31>HTPR** and bringing the input **TestPEHLPR** high (enabling **D<0:31>HTPR** as output drivers), and then bringing the input **TestOEHLPR** high at least one cycle later (enablingthe microcode memory outputs onto **D<0:31>HSP2**). Several cycles later the data from the addressed location is valid on **D<0:31>HTPR**. Before addressing a new location, **TestOEHLPR** must first be brought low, and then **TestPEHLPR** brought low at least one cycle later. The inputting device (i.e.: chip tester) must tri-state **D<0:31>HTPR** whenever **TestPEHLPR** is asserted. For normal operation, **TestOEHLPR** and **TestPEHLPR** should be grounded..