HOLT INTEGRATED CIRCUITS

AN-555

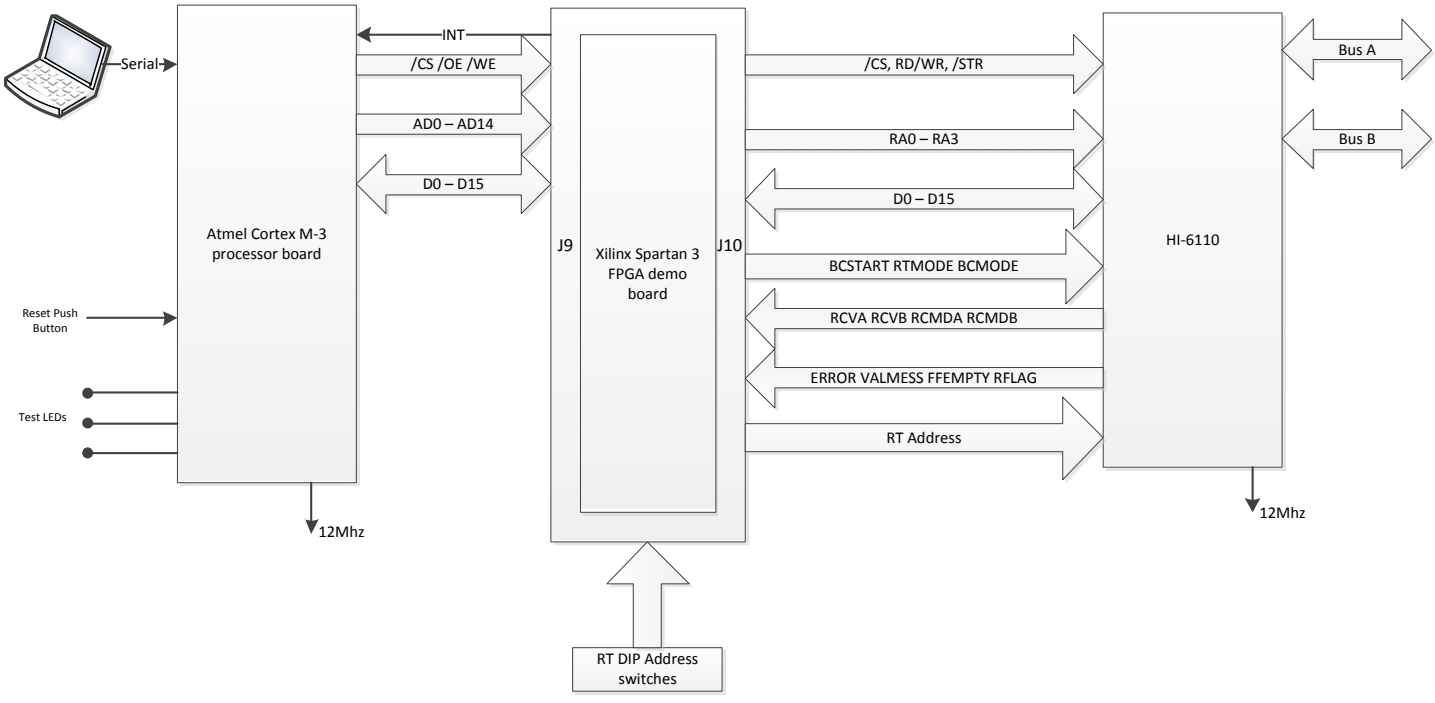
HI-6110 RT FPGA Integration Application Note

May 4, 2012

Introduction

This application note demonstrates how to implement a MIL-STD-1553 remote terminal (RT) using an HI-6110 single message processor managed by a field-programmable gate array (FPGA). The provided Verilog RTL implements RAM buffers within the FPGA for all 30 receive subaddresses, all 30 transmit subaddresses and all mode commands with data word. Most importantly, the FPGA fully handles all required real-time support for the HI-6110 RT, including illegal command detection, RT status maintenance, bus switching, time-tag counter, error handling and required mode command responses.

The reference design includes an ARM Cortex M3 microprocessor that accesses FPGA RAM using a dedicated 16-bit parallel bus. The primary purpose of this ARM MCU is design verification: loading FPGA RAM buffers for transmit subaddresses, offloading FPGA RAM buffers for receive subaddresses, and accepting message interrupts generated by Verilog RTL. The MCU is programmed to provide RS-232 console output for display by a PC terminal like HyperTerminal. This MCU console output displays transacted message data and debug information. **A typical application does not include the design-verification MCU. Instead, the Verilog RTL is expanded to encompass all required Remote Terminal functions.**



Documents and support files

The documentation and design files are provided in two zip files.

6110RT_FPGA_1.ZIP (3.016 Meg)

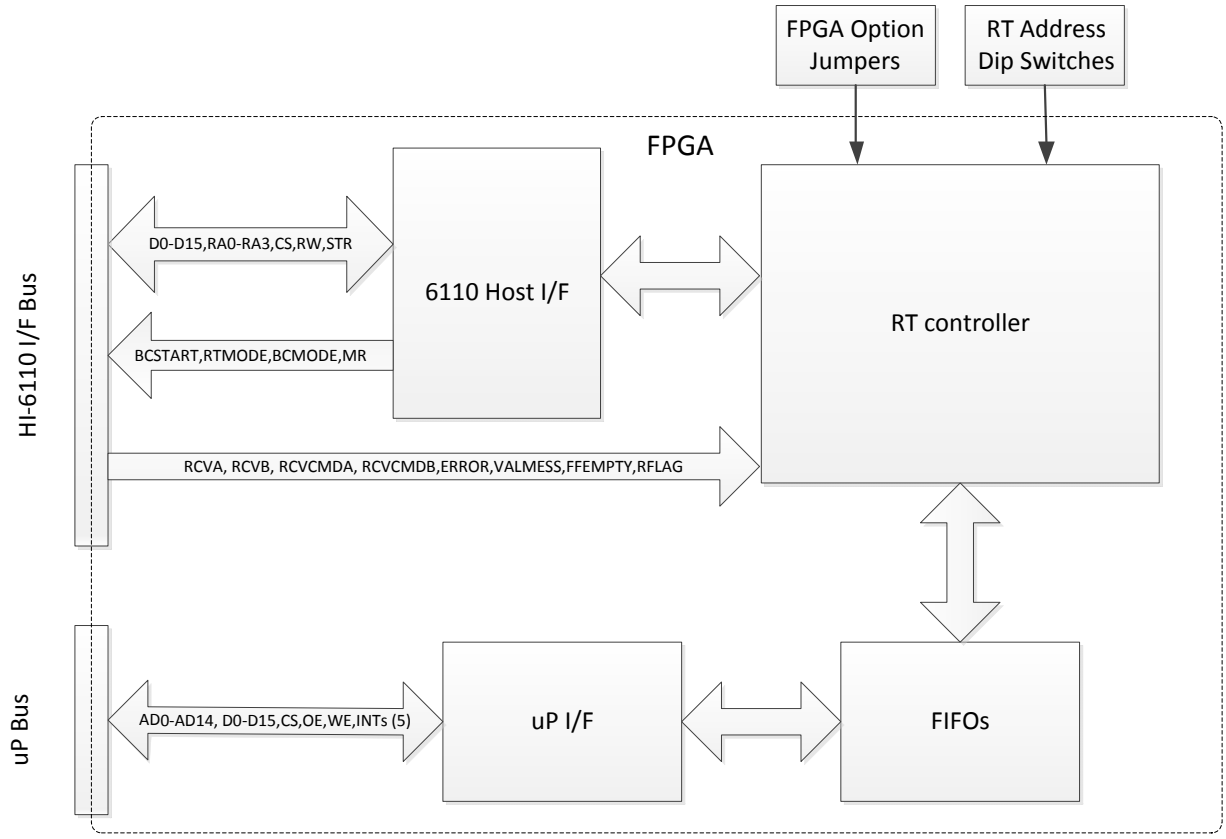
Item	Description
AN-555 This application note	This Application Note
HI-6110 Data Sheet	Data Sheet
_Holt HI-6110 FPGA Demo 0_4.zip	IAR EWB project files
blk_mem_gen_ds512.pdf	XILINX FPGA Product Specification
fifomap.xlsx	FIFO memory map spreadsheet

6110RT_FPGA_2.ZIP (8.5 Meg)

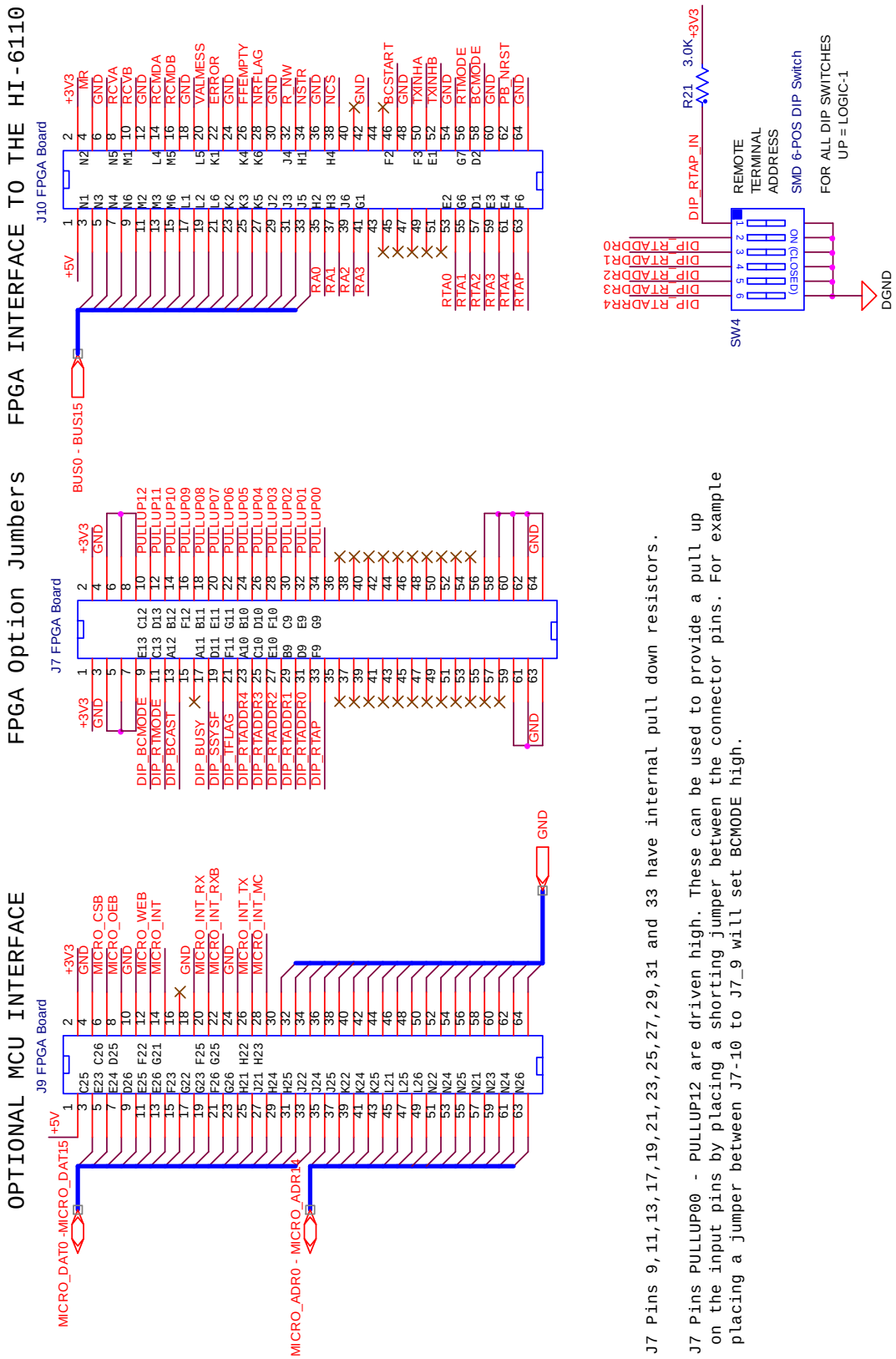
Item	Description
FPGA folder	FPGA Verilog source files
CHIP folder	FPGA related design files
Readme	Readme text file
fpga_6110_summary.html	FPGA logic utilization

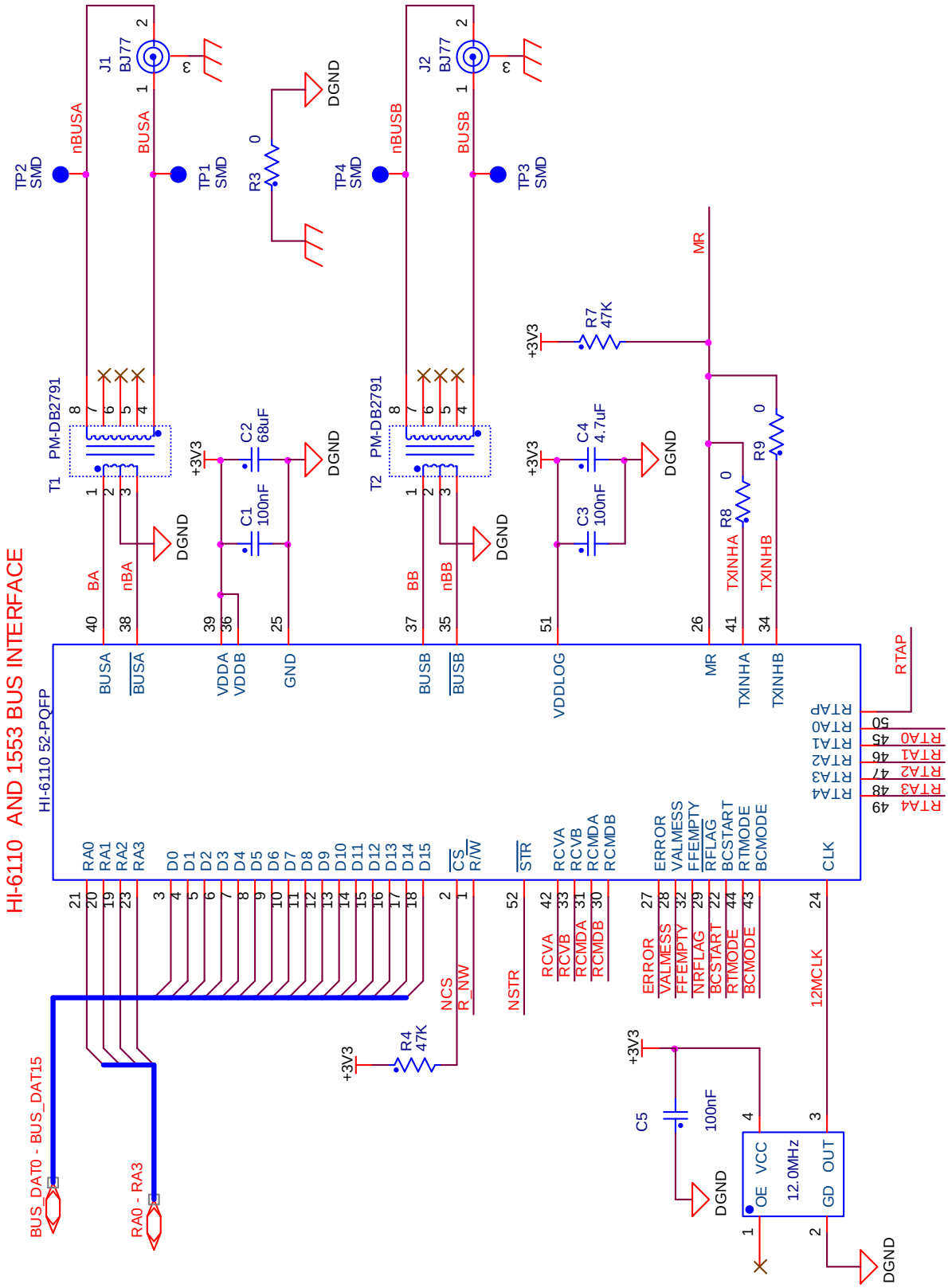
Holt provides these documents and design files as a design guide how to implement the HI-6110 MIL-STD 1553 protocol IC into a FPGA design. No demo board is currently available.

FPGA Functional Block Diagram



Simplified Schematic Diagram – Orange Tree FPGA Board Connectors




HI-6110 AND 1553 BUS INTERFACE
HI-6110 52-PQFP
J1 BJ77
J2 BJ77
TP1 SMD
TP2 SMD
TP3 SMD
TP4 SMD
T1 PM-DB2791
T2 PM-DB2791
R3 0
R4 47K
R7 47K
R8 0
R9 0
C1 100nF
C2 68uF
C3 100nF
C4 4.7uF
C5 100nF
DGND
+3V3
nBUSA
BUSA
nBUSB
BUSB
BA
nBA
BB
nBB
VDDA
VDDB
GND
VDDLOG
MR
TXINHA
TXINHB
RTAP
RTA0
RTA1
RTA2
RTA3
RTA4
RA0
RA1
RA2
RA3
D0
D1
D2
D3
D4
D5
D6
D7
D8
D9
D10
D11
D12
D13
D14
D15
CS
R/W
STR
RCVA
RCVB
RCMDA
RCMDB
ERROR
VALMESS
FFEMPTY
RFLAG
BCSTART
RTMODE
BCMODE
CLK
NCS
R_NW
NSTR
12MCLK
12.0MHz
OE
VCC
GD
OUT
BUS_DAT0 - BUS_DAT15
RA0 - RA3

The FPGA memory map:

The FPGA memory map is addressed using fifteen address lines AD0-AD14 and a 16-bit data bus D0-D15. The full address range for the FPGA memory map is 0 - 0x41A2. This is mapped on the processor side starting at 0x6000-0000.

FPGA map address range	Cortex M-3 Processor address range
0 - 0x41A2	0x60000000 - 0x600041A2

The memory map is segmented by the message sub address (SA) and by the type of the message that was received.

FPGA memory map table summary:

Type	Address	Comments
Rx Message data	0 – 0x27FF	Rx Message FIFO area. 32 words x 30.
RxB Message data	0x800 – 0x2FFF	Rx Broadcast FIFO area.32 words x 30.
TX Message data	0x1000 – 0x37FF	Transmit data FIFO area. 32 words x 30.
Rx Mode Code data	0x1800 – 0x38FF	Rx Mode Code FIFO area. Mode code 16-31.
Rx Broadcast Mode Code data	0x1A00 – 0x3AFF	Rx Broadcast Mode Code FIFO. Mode codes 16-31
Tx Mode Code data	0x1c00 – 0x3DF0	Transmit Mode Code FIFO. 0-31
TxBroadcast Mode Code data	0x1E00 – 0x3EFF	Transmit Broadcast Mode Code FIFO.
Interrupt Flags by SA	0x4000 – 0x4014	Interrupt Flag bits indicating a received message by SA.
Time Tag register (16-bits)	0x401E	Time Tag. 64us increment rate.
Interrupt Enables by SA	0x4020 – 0x4034	Interrupt enable by SA and by type.
FIFO Status addresses	0x4040 – 0x40FE	Status register by SA for Non Mode Code messages.
Mode Code Status addresses	0x4100 – 0x419E	Status register by SA for Mode Code messages.
FIFO Status2	0x4240 – 42FE	Additional status register providing Bus A/B flag.
Current Command Word	0x41A0	Register contains current command word.
Last Command Word	0x41A2	Register contains last command word.

Sub-address transmit or receive commands (non mode codes) can have up to 32 data words. For a receive command, the FPGA stores up to 32 data words into unique addresses for each of the possible 30 SA's. The FPGA alternately stores messages in either an X bank or Y bank in a ping-pong fashion. The Rx X bank flag (D14) in the status register indicates which bank was written to last that was received. The Rx bank flag bit in the status register alternates between a 0 and a 1 for each received message. The status register provides information about the message as follows.

Status Register:

D15 – RT-RT flag. Logic-1 indicates an RT-RT message. Logic-0 would indicate a non RT-RT message.

D14 – Rx Bank. A 1 indicates bank X. A 0 indicates bank Y.

D13 – Valid Message X. If D14 was 1, then this bit is checked for a valid status. Valid status=1.

D12 – Valid Message Y. If D14 was 0, then this bit is check for valid status. Valid status=1.

D11 – Error Message X. If the message had an error this bit will be 1. Valid when D14=1.

D10 - Error Message Y. If the message had an error this bit will be 1. Valid when D14=0.

D5-D9 Word Count for X bank.

D0-D4 Word Count for Y bank.

The following example shows how the FPGA stores a message for a receive command on sub-address (SA)=1.

All the other messages work similarly.

1. Enable Receive messages on SA-1 by setting D1 high in the Rx Interrupt Enable location 0x4020. If you wanted to also receive other messages with a different SA then set those bits high also in this step. The demo program default sets all the bits high so all SA messages will be received.

2. Poll the Rx Interrupt Flags at location 0x4000. A simple polling scheme can be used to detect if a new message has arrived or an interrupt can be used to schedule the polling. The interrupt output of the FPGA is a global signal. It will pulse for 5.4us when any enabled interrupt occurs. There is no action needed by the processor to clear the interrupt signal since it is a pulse.

3. When a receive message arrives on SA=1 the FPGA sets address location 0x4000 bit D1 high. Logic-1 on D1 reflects the SA of the message. This SA is used as an index into the corresponding status registers and FIFO data word registers for this message. First, a pointer is calculated by adding the base address 0x60000000 to the SA index value. Fixed values are added to this pointer value to get the status register location and the starting location where the 32 data words are stored. The program determines whether the X or Y banks should be referenced, the status of the Valid Message and Error flags in the status register and the word count. The demo code demonstrates all this by examining these bits and displaying the results on the console.

4. These are the corresponding locations for this example with a receive command a SA=1:

   Interrupt enable flag address=0x4020, D1 bit.

   Rx Interrupt Flag address = 0x4000, D1 bit.

   Status Register address = 0x4040. If SA = 2 then the status register location is 0x4042.

   Status Register2 = 0x4240. This is where the A or B flags are stored to indicate which bus the message arrived from.

   FIFO start data word addresses (SA=1) 0x0040 for bank X or 0x2040 for bank Y.

   If for example this was a Transmit message then the FIFO start data word address would begin at 0x1000 for the X bank.

For transmit commands, the FPGA retrieves the data stored at the TX FIFO locations and transmits those values. The FPGA uses the SA and word count fields in the received message to determine the FIFO data locations to use to compile the transmit command sent back to the BC. The demo code initializes the TX FIFO locations with unique arbitrary data patterns so that the data can be easily identified when a transmit message is transmitted on the bus. This is strictly for demo purposes.

For example if the SA=3 and the word count was 3 then a TX message using bank X would contain the three data words 0x0300, 0x0301, 0x0302. If the same transmit message is received again the data words transmitted would be 0x2300, 0x2301, 0x2302 from bank Y. The 3 in the upper byte and lower nibble reflects the SA and the 2 in the upper byte and upper nibble indicates the bank. The lower 8 bits contain the word count.

If a 32 word transmit message is received the FPGA would transmitted 32 words from the X bank:

```
0x0300 0x0301 0x0302 0x0303 0x0304 0x0305 0x0306 0x0307
0x0308 0x0309 0x030A 0x030B 0x030C 0x030D 0x030E 0x030F
0x0310 0x0311 0x0312 0x0313 0x0314 0x0315 0x0316 0x0317
0x0318 0x0319 0x031A 0x031B 0x031C 0x031D 0x031E 0x031F
```

Refer to the tables 1, 2 and 3 for all the FPGA FIFO address locations.

Time Tag:

The FPGA automatically increments a free-running 16-bit time tag counter every 64us and stores this value at 0x401E. The microcontroller can read this value any time. If the RT receives a Transmit Mode Code 1 command, the value will be cleared. If a Receive Mode Code 17 (synchronize with data) is received, the timer is loaded with the mode data word from Mode Code command.

HI-6110 interface:

The FPGA interface to the HI-6110 monitors the signals from the HI-6110 and responds to MIL-STD 1553 messages automatically. The FPGA performs all low-level servicing of the HI-6110 to detect,

qualify and store the resulting messages into the FIFO memory locations. All the logic to reject errors and handle superseding messages and bus switching are fully managed by the HI-6110 and FPGA. To learn more about this side of the interface refer to the HI-6110 data sheet and the FPGA Verilog code.

The addresses shown in the following tables are referenced to the FPGA. The microcontroller maps this space at 0x60000000.

Table 1 Command RAM Buffers.

Receive Commands

RX subadr	Bank X adr	Bank Y adr
1	40	2040
2	80	2080
3	C0	20C0
4	100	2100
5	140	2140
6	180	2180
7	1C0	21C0
8	200	2200
9	240	2240
10	280	2280
11	2C0	22C0
12	300	2300
13	340	2340
14	380	2380
15	3C0	23C0
16	400	2400
17	440	2440
18	480	2480
19	4C0	24C0
20	500	2500
21	540	2540
22	580	2580
23	5C0	25C0
24	600	2600
25	640	2640
26	680	2680
27	6C0	26C0
28	700	2700
29	740	2740
30	780	2780

Receive Broadcast Commands

RXB subadr	Bank X adr	Bank Y adr
1	840	2840
2	880	2880
3	8C0	28C0
4	900	2900
5	940	2940
6	980	2980
7	9C0	29C0
8	A00	2A00
9	A40	2A40
10	A80	2A80
11	AC0	2AC0
12	B00	2B00
13	B40	2B40
14	B80	2B80
15	BC0	2BC0
16	C00	2C00
17	C40	2C40
18	C80	2C80
19	CC0	2CC0
20	D00	2D00
21	D40	2D40
22	D80	2D80
23	DC0	2DC0
24	E00	2E00
25	E40	2E40
26	E80	2E80
27	EC0	2EC0
28	F00	2F00
29	F40	2F40
30	F80	2F80

Transmit Commands

TX subadr	Bank X adr	Bank Y adr
1	1040	3040
2	1080	3080
3	10C0	30C0
4	1100	3100
5	1140	3140
6	1180	3180
7	11C0	31C0
8	1200	3200
9	1240	3240
10	1280	3280
11	12C0	32C0
12	1300	3300
13	1340	3340
14	1380	3380
15	13C0	33C0
16	1400	3400
17	1440	3440
18	1480	3480
19	14C0	34C0
20	1500	3500
21	1540	3540
22	1580	3580
23	15C0	35C0
24	1600	3600
25	1640	3640
26	1680	3680
27	16C0	36C0
28	1700	3700
29	1740	3740
30	1780	3780

Mode Code commands

MC RX mcode	Bank X adr	Bank Y adr
16	1800	3800
17	1810	3810
18	1820	3820
19	1830	3830
20	1840	3840
21	1850	3850
22	1860	3860
23	1870	3870
24	1880	3880
25	1890	3890
26	18A0	38A0
27	18B0	38B0
28	18C0	38C0
29	18D0	38D0
30	18E0	38E0
31	18F0	38F0

MC RXB mcode	Bank X adr	Bank Y adr
16	1A00	3A00
17	1A10	3A10
18	1A20	3A20
19	1A30	3A30
20	1A40	3A40
21	1A50	3A50
22	1A60	3A60
23	1A70	3A70
24	1A80	3A80
25	1A90	3A90
26	1AA0	3AA0
27	1AB0	3AB0
28	1AC0	3AC0
29	1AD0	3AD0
30	1AE0	3AE0
31	1AF0	3AF0

MC TX mcode	Bank X adr	Bank Y adr
0	1C00	3C00
1	1C10	3C10
2	1C20	3C20
3	1C30	3C30
4	1C40	3C40
5	1C50	3C50
6	1C60	3C60
7	1C70	3C70
8	1C80	3C80
9	1C90	3C90
10	1CA0	3CA0
11	1CB0	3CB0
12	1CC0	3CC0
13	1CD0	3CD0
14	1CE0	3CE0
15	1CF0	3CF0
16	1D00	3D00
17	1D10	3D10
18	1D20	3D20
19	1D30	3D30
20	1D40	3D40
21	1D50	3D50
22	1D60	3D60
23	1D70	3D70
24	1D80	3D80
25	1D90	3D90
26	1DA0	3DA0
27	1DB0	3DB0
28	1DC0	3DC0
29	1DD0	3DD0
30	1DE0	3DE0
31	1DF0	3DF0

MC TXB mcode	Bank X adr	Bank Y adr
0	1E00	3E00
1	1E10	3E10
2	1E20	3E20
3	1E30	3E30
4	1E40	3E40
5	1E50	3E50
6	1E60	3E60
7	1E70	3E70
8	1E80	3E80
9	1E90	3E90
10	1EA0	3EA0
11	1EB0	3EB0
12	1EC0	3EC0
13	1ED0	3ED0
14	1EE0	3EE0
15	1EF0	3EF0

Receive, Receive Broadcast and Transmit interrupt flags.

Address (hex) RegBits	4000	4002	4004	4006	4008	400A
15	Rx Int Flag[15]	Rx Int Flag[31]	RxB Int Flag[15]	RxB Int Flag[31]	Tx Int Flag[15]	Tx Int Flag[31]
14	14	30	14	30	14	30
13	13	29	13	29	13	29
12	12	28	12	28	12	28
11	11	27	11	27	11	27
10	10	26	10	26	10	26
9	9	25	9	25	9	25
8	8	24	8	24	8	24
7	7	23	7	23	7	23
6	6	22	6	22	6	22
5	5	21	5	21	5	21
4	4	20	4	20	4	20
3	3	19	3	19	3	19
2	2	18	2	18	2	18
1	1	17	1	17	1	17
0	0	16	0	16	0	16

Receive, Receive Broadcast and Transmit interrupt Enable flags.

Address (hex) RegBits	4020	4022	4024	4026	4028	402A
15	Rx Int En[15]	Rx Int En[31]	RxB Int En[15]	RxB Int En[31]	Tx Int En[15]	Tx Int En[31]
14	14	30	14	30	14	30
13	13	29	13	29	13	29
12	12	28	12	28	12	28
11	11	27	11	27	11	27
10	10	26	10	26	10	26
9	9	25	9	25	9	25
8	8	24	8	24	8	24
7	7	23	7	23	7	23
6	6	22	6	22	6	22
5	5	21	5	21	5	21
4	4	20	4	20	4	20
3	3	19	3	19	3	19
2	2	18	2	18	2	18
1	1	17	1	17	1	17
0	0	16	0	16	0	16

Mode Code interrupt and interrupt enable flags.

400C	400E	4010	4012	4014
MC Rx Int Flag[31]	MC RxB Int Flag[31]	MC Tx Int Flag[15]	MC Tx Int Flag[31]	MC TxB Int Flag[15]
30	30	14	30	14
29	29	13	29	13
28	28	12	28	12
27	27	11	27	11
26	26	10	26	10
25	25	9	25	9
24	24	8	24	8
23	23	7	23	7
22	22	6	22	6
21	21	5	21	5
20	20	4	20	4
19	19	3	19	3
18	18	2	18	2
17	17	1	17	1
16	16	0	16	0

402C	402E	4032	4030	4034
MC Rx Int En[31]	MC RxB Int En[31]	MC Tx Int En[15]	MC Tx Int En[31]	MC TxB Int En[15]
30	30	14	30	14
29	29	13	29	13
28	28	12	28	12
27	27	11	27	11
26	26	10	26	10
25	25	9	25	9
24	24	8	24	8
23	23	7	23	7
22	22	6	22	6
21	21	5	21	5
20	20	4	20	4
19	19	3	19	3
18	18	2	18	2
17	17	1	17	1
16	16	0	16	0

Receive, Receive Broadcast and Transmit Status Registers.

Address (hex) RegBits	4042 – 407C SA 1 – SA30		4082 - 40BC SA 1 – SA30		40C2 - 40FC SA 1 – SA30	
15	RTRT flag	RTRT flag	RTRT flag	RTRT flag	RTRT flag	RTRT flag
14	Rx 1 Bank	Rx 30 Bank	RxB 1 Bank	RxB 30 Bank	Tx 1 Bank	Tx 30 Bank
13	Val Mess X Rx 1	Val Mess X Rx 30	Val Mess X RxB 1	Val Mess X RxB 30	Val Mess X Tx 1	Val Mess X Tx 30
12	Val Mess Y Rx 1	Val Mess Y Rx 30	Val Mess Y RxB 1	Val Mess Y RxB 30	Val Mess Y Tx 1	Val Mess Y Tx 30
11	Err Mess X Rx 1	Err Mess X Rx 30	Err Mess X RxB 1	Err Mess X RxB 31	Err Mess X Tx 1	Err Mess X Tx 30
10	Err Mess Y Rx 1	Err Mess Y Rx 30	Err Mess Y RxB 1	Err Mess Y RxB 30	Err Mess Y Tx 1	Err Mess Y Tx 30
9	Wd Cnt X Rx 1 [4]	Wd Cnt X Rx 30 [4]	Wd Cnt X RxB 1 [4]	Wd Cnt X RxB 30 [4]	Wd Cnt X Tx1 [4]	Wd Cnt X Tx 30 [4]
8	3	3	3	3	3	3
7	2	2	2	2	2	2
6	1	1	1	1	1	1
5	0	0	0	0	0	0
4	Wd Cnt Y Rx 1[4]	Wd Cnt Y Rx 30 [4]	Wd Cnt Y RxB 1 [4]	Wd Cnt Y RxB 30 [4]	Wd Cnt Y Tx 1 [4]	Wd Cnt Y Tx 30 [4]
3	3	3	3	3	3	3
2	2	2	2	2	2	2
1	1	1	1	1	1	1
0	0	0	0	0	0	0

Example for Receive commands with SA=1,2 and 30.

SA=1, Address = 0x4042.

SA=2, Address = 0x4044.

.
.
.

SA=30, Address = 0x407C.

Receive, Receive Broadcast and Transmit Status Registers for Bus A/B Status Registers.

Address (hex) RegBits	4242 – 427C SA 1 – SA30		4280 - 42BE SA 1 – SA30		42C0 - 42FE SA 1 – SA30	
15	-	-	-	-	-	-
14	-	-	-	-	-	-
13	-	-	-	-	-	-
12	-	-	-	-	-	-
11	-	-	-	-	-	-
10	-	-	-	-	-	-
9	-	-	-	-	-	-
8	-	-	-	-	-	-
7	-	-	-	-	-	-
6	-	-	-	-	-	-
5	-	-	-	-	-	-
4	-	-	-	-	-	-
3	Bus A X Rx 1 active	Bus A X Rx 30 active	Bus A X RxB 1active	Bus A X RxB 30active	Bus A X Tx 1active	Bus A X Tx 30 active
2	Bus A Y Rx 1 active	Bus A Y Rx 30 active	Bus A Y RxB 1 active	Bus A Y RxB 30 active	Bus A Y Tx1 active	Bus A Y Tx 30 active
1	Bus B X Rx 1active	Bus B X Rx 30 active	Bus B X RxB 1active	Bus B X RxB 30 active	Bus B X Tx 1 active	Bus B X Tx 30active
0	Bus B Y Rx 1 active	Bus B Y Rx 30 active	Bus B Y RxB 1 active	Bus B Y RxB 30 active	Bus B Y Tx 1 active	Bus B Y Tx 30 active

Example for Receive commands with SA=1,2 and 30.

SA=1, Address = 0x4242.

SA=2, Address = 0x4244.

.
.
.

SA=30, Address = 0x427C.

Mode Code Status Registers.

Address (hex) RegBits	4100	411E	4120	413E	4140	417E	4180	419E
15	-	-	-	-	-	-	-	-
14	MC Rx 16 Bank	MC Rx 31 Bank	MC RxB 16 Bank	MC RxB 31 Bank	MC Tx 0 Bank	MC Tx 31 Bank	MC TxB 0 Bank	MC TxB 15 Bank
13	Val Mess X MC Rx 16	Val Mess X MC Rx 31	Val Mess X MC RxB 16	Val Mess X MC RxB 31	Val Mess X MC Tx 0	Val Mess X MC Tx 31	Val Mess X MC TxB 0	Val Mess X MC TxB 15
12	Val Mess Y MC Rx 16	Val Mess Y MC Rx 31	Val Mess Y MC RxB 16	Val Mess Y MC RxB 31	Val Mess Y MC Tx 0	Val Mess Y MC Tx 31	Val Mess Y MC TxB 0	Val Mess Y MC TxB 15
11	Err Mess X MC Rx 16	Err Mess X MC Rx 31	Err Mess X MC RxB 16	Err Mess X MC RxB 31	Err Mess X MC Tx 0	Err Mess X MC Tx 31	Err Mess X MC TxB 0	Err Mess X MC TxB 15
10	Err Mess Y MC Rx 16	Err Mess Y MC Rx 31	Err Mess Y MC RxB 16	Err Mess Y MC RxB 31	Err Mess Y MC Tx 0	Err Mess Y MC Tx 31	Err Mess Y MC TxB 0	Err Mess Y MC TxB 15
9	-	-	-	-	-	-	-	-
8	-	-	-	-	-	-	-	-
7	-	-	-	-	-	-	-	-
6	-	-	-	-	-	-	-	-
5	-	-	-	-	-	-	-	-
4	-	-	-	-	-	-	-	-
3	Bus A X MC Rx 16 actv	Bus A X MC Rx 31 actv	Bus A X MC RxB 16 actv	Bus A X MC RxB 31 actv	Bus A X MC Tx 0 actv	Bus A X MC Tx 31 actv	Bus A X MC TxB 0 actv	Bus A X MC TxB 15 actv
2	Bus A Y MC Rx 16 actv	Bus A Y MC Rx 31 actv	Bus A Y MC RxB 16 actv	Bus A Y MC RxB 31 actv	Bus A Y MC Tx 0 actv	Bus A Y MC Tx 31 actv	Bus A Y MC TxB 0 actv	Bus A Y MC TxB 15 actv
1	Bus B X MC Rx 16 actv	Bus B X MC Rx 31 actv	Bus B X MC RxB 16 actv	Bus B X MC RxB 31 actv	Bus B X MC Tx 0 actv	Bus B X MC Tx 31 actv	Bus B X MC TxB 0 actv	Bus B X MC TxB 15 actv
0	Bus B Y MC Rx 16 actv	Bus B Y MC Rx 31 actv	Bus B Y MC RxB 16 actv	Bus B Y MC RxB 31 actv	Bus B Y MC Tx 0 actv	Bus B Y MC Tx 31 actv	Bus B Y MC TxB 0 actv	Bus B Y MC TxB 15 actv

Command Words

The Current Command Word is a copy of the actual command word received by the HI-6110. The Last Command Word is the previous command.

Address (hex) RegBits	41A0	41A2
15	Current CW[15]	Last CW[15]
14	14	14
13	13	13
12	12	12
11	11	11
10	10	10
9	9	9
8	8	8
7	7	7
6	6	6
5	5	5
4	4	4
3	3	3
2	2	2
1	1	1
0	0	0

FPGA Option Jumpers

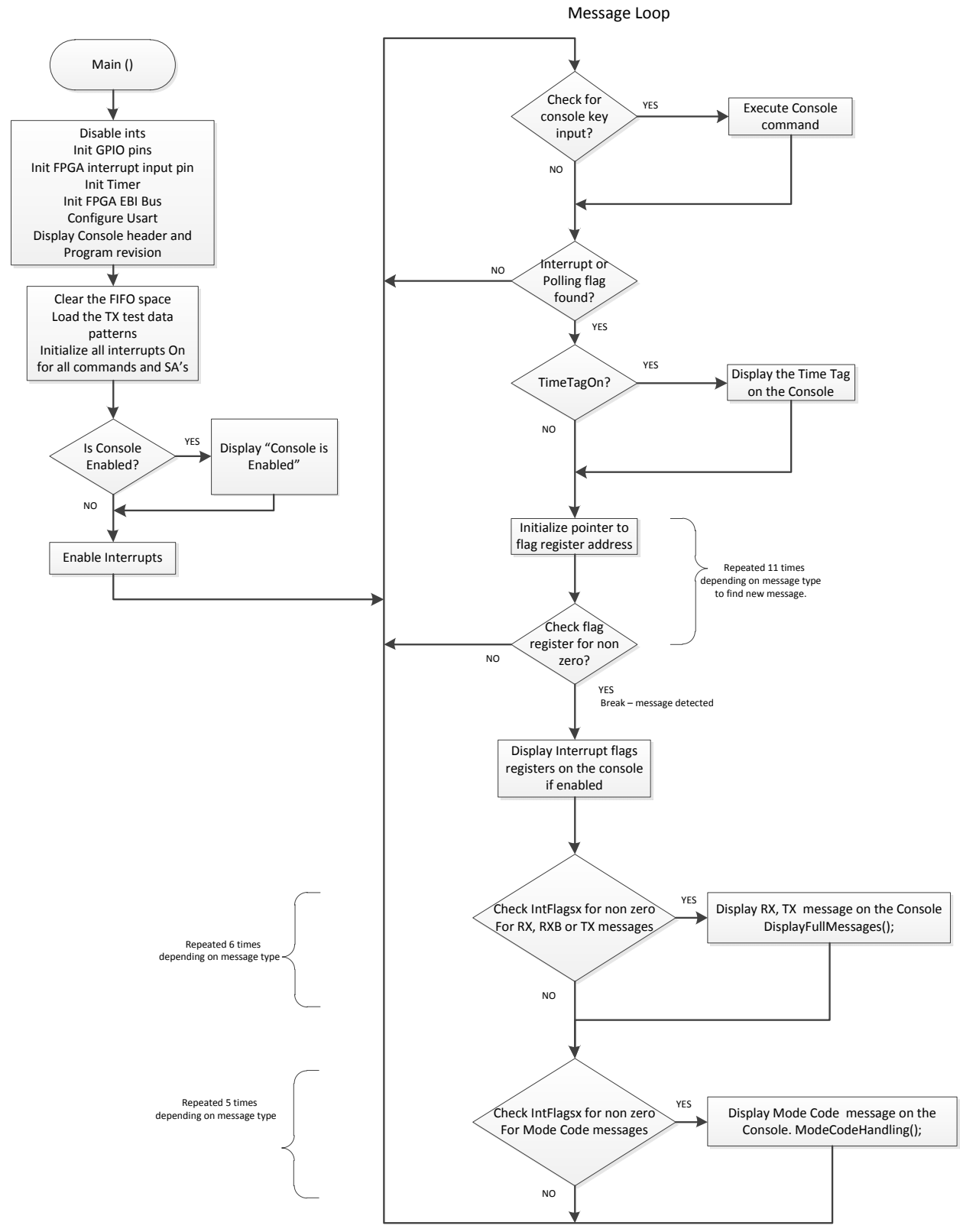
These optional jumpers affect broadcast command response as well as the MIL-STD 1553 Status Word (SW). Refer to the MIL-STD 1553 specification for more details of these bits and their use. The default configuration is with open jumpers.

Function	Orange Tree FPGA board Connector Pin	FPGA assignment	Configuration
BCMODE	J7-9	E13	Input, pass through to 6110 BCMODE input. Set low.
RTMODE	J7-11	C13	Input, pass through to 6110 RTMODE input. Set high.
Disable Broadcast Commands	J7-13	A12	Disable=Closed, Enable=Open
Force BUSY Status bit high	J7-17	A11	Closed forces the BUSY flag bit high in the Status Word
Force SUBSYSTEM FLAG high	J7-19	D11	Closed forces the SUBSYSTEM flag bit high the Status Word
Force TERMINAL FLAG Status bit high	J7-21	F11	Closed forces the Terminal Flag bit high in the Status Word.

Host MCU Demo Software (optional)

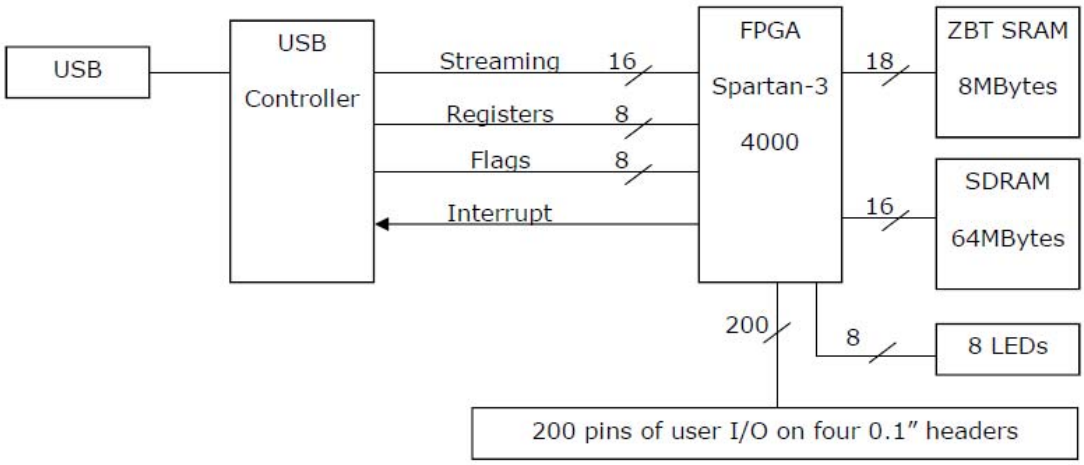
The microcontroller interface to the FPGA allows messages to be enabled and received. The application may not need this capability but is provided as a means to verify the operation of the FPGA interface to the HI-6110.

The MCU demo software is programmed in the flash memory in the Atmel Cortex M-3 microcontroller so the demo program can be used immediately after powering up the board. IAR's KickStart version of the Embedded Workbench® IDE toolset version 6.2 was used to develop this software. See the end of this application note for more details on the toolset and program files for this project. Software flow charts are provided to understand the overall flow of the program.

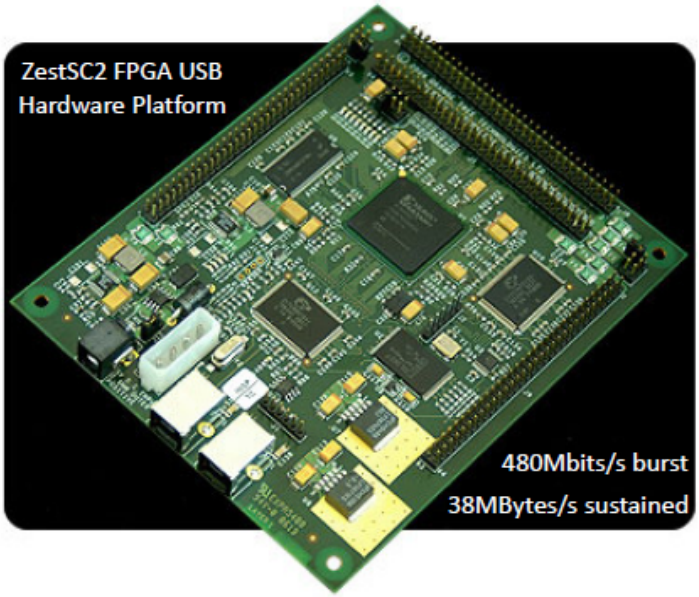

Message Loop
Main ()
Disable ints
Init GPIO pins
Init FPGA interrupt input pin
Init Timer
Init FPGA EBI Bus
Configure Usart
Display Console header and Program revision
Clear the FIFO space
Load the TX test data patterns
Initialize all interrupts On for all commands and SA's
Is Console Enabled?
YES
Display "Console is Enabled"
NO
Enable Interrupts
Check for console key input?
YES
Execute Console command
NO
Interrupt or Polling flag found?
NO
YES
TimeTagOn?
YES
Display the Time Tag on the Console
NO
Initialize pointer to flag register address
Repeated 11 times depending on message type to find new message.
Check flag register for non zero?
NO
YES
Break – message detected
Display Interrupt flags registers on the console if enabled
Repeated 6 times depending on message type
Check IntFlagsx for non zero For RX, RXB or TX messages
YES
Display RX, TX message on the Console DisplayFullMessages();
NO
Repeated 5 times depending on message type
Check IntFlagsx for non zero For Mode Code messages
YES
Display Mode Code message on the Console. ModeCodeHandling();
NO

FPGA Board

An Orange Tree Technologies ZestSC2 FPGA development board is used in this example. For more information on this board you can contact Orange Tree Technologies directly.



ZestSC2: High specification prototyping and development solution



Xilinx Spartan 3 FPGA
•3S2000/ 3S4000

USB
•Cypress EZ-USB FX2

Memory
•8MB pipelined ZBT RAM
•32MB/ 64MB SDRAM
•Flash memory for FPGA configuration

•200 pins of user I/O

•SRAM, SDRAM and USB interface logic cores

•Self powered, wall adaptor, secondary USB port (5W) or hard disc power connector

•Host library and drivers (WinXP/ Linux)

•Reference designs inc. VHDL/ Verilog source

Email: info@orangetreetech.com

Visit: www.orangetreetech.com

REVISION HISTORY

P/N	Rev	Date	Description of Change
AN-555	New	5/4/12	Initial Release