Using the FADC125 Module (V1 - 5/21/14)

1.1 Controlling the Module

Communication with the module is by standard VME bus protocols. All registers and memory locations are defined to be 4-byte entities. The VME slave module has three distinct address ranges.

A24 – The base address of this range is set by a 12-element DIP switch on the board. It occupies 4 Kbytes of VME address space, organized in 1 K 32-bit words. Relative to the base address, this space is divided as follows:

0000-0FFF – Register space for Main (VME) FPGA

1000-CFFF – Register space for Front End ADC FPGAs
12 total - 4th word represents 1-12 (0x1-0xC)

D000-FFFF – Register space for Processor FPGA

A32 - The base address of this range is programmed into register ADR32. It occupies 8 Mbytes of VME address space, organized in 2 M 32-bit words. A read of any address in this range will yield the next FADC data word from the module. Even though the module is logically a FIFO, the expanded address range allows the VME master to increment the address during block transfers. This address range can participate in single cycle, 32-bit block, and 64-bit block reads. The only valid write to this address range is the data value 0x80000000 which re-enables the module to generate interrupts (after one has occurred). The address range must be enabled by setting ADR32[0] = 1.

A32 - The lower and upper limits of this address range are programmed into register ADR_MB. This common address range for a set of FADC modules in the crate is used to implement the Multiblock protocol. By means of token passing FADC data may be read out from multiple FADC modules using a single logical block read. The board possessing the token will respond to a read cycle in this address range with the next FADC data word from that module. The token is passed along a private daisy chain line to the next module when it has transferred all data from a programmed number of events (register BLOCK SIZE). The address range must be enabled by setting ADR_MB[0] = 1.

1.2 Module Registers

fADC125 register map
test firmware: Raw mode, PTW
Firmware version 0x0001 – 0x101

Main FPGA(0x0000-0x0ffc)

Board ID – Status (0x0000)
0xADC12500

Swap Control – Control/Status (0x0004)
Any write other than "0" swaps.

Firmware version – Status (0x0008)

Main CSR - Control/Status (0x000C)
1-0 (R/W) – select clock
00 – P2 clock
01 – P0 clock
10 – On
11 – local 125Mhz – (default in firmware)

Power Control - Control/Status (0x0010)
Writing 0x3000ABCD turns "ON",
Anything else turns "OFF"

DAC Control – Control (0x0014) – note: write only
0 – (W) –
1 – (W) – Serial Interface Chip Load (main and mezz chains)
2 – (W) – Serial Interface Clock Input (main and mezz chains)
3 – (W) –
4 – (W) – Serial Interface Data Input (main chain)
5 – (W) –
6 – (W) –
7– (W) –
8 – (W) – Serial Interface Data Input (mezz chain)

Control/Status (0x0018 -0x001C) – used for testing

Serial – Status (0x0020-0x002C)
0x0020 – main_serial(47 downto 32)
0x0024 – main_serial(31 downto 0)

0x0028 – mezz_serial(47 downto 32)
0x002C – mezz_serial(31 downto 0)

Temperature – Status (0x0030-0x0034)

0x0030 – main temperature
0x0034 – mezz temperature

Geographical slot address – Status (0x0038)

0 – (R) – bit 0 of GAD
1 – (R) – bit 1 of GAD
2 – (R) – bit 2 of GAD
3 – (R) – bit 3 of GAD
4 – (R) – bit 4 of GAD

A32 base address – Status (0x003C)

Block CSR – Control/Status (0x0040)

0 – (R) –
1 – (R) –
2 – (R) – Block of Events ready for readout
3 – (R) – BERR Status (1 = BERR asserted)
4 – (R) – Token Status (1 = module has token)
5 – (W) – Take Token
6 – (W) – Pulse Soft Sync Reset
7 – (W) – Pulse Soft Trigger (ACTUALLY bit 0 of 0xd010, on proc)
8 – (W) – Pulse Soft Reset
9 – (W) – Pulse Hard Reset

CTRL1 – Control/Status (0x0044)

1-0 (R/W) – Sync reset source select N/A, MOVED TO PROC (0xd00C)
00– (R/W) – N/A, moved to proc
01 – (R/W) – N/A, moved to proc
10 – (R/W) – N/A, moved to proc
11 – (R/W) – N/A, moved to proc
2 – (R/W) – Enable BERR response
3 – (R/W) – Enable Multiboard protocol
4 – (R/W) – FIRST board in Multiblock system
5 – (R/W) – LAST board in Multiblock system

ADR32 – Control/Status (0x0048)

0 – (R/W) – Enable 32-bit address decoding
[1… 6] – (not used – read as 0)
[15…7] – (R/W) – Base Address for 32-bit addressing mode (8 Mbyte total)

ADR_MB – Multiblock Address for data access (0x004C)

0 – (R/W) – Enable Multiblock address decoding

1 – 6 – (not used – read as 0)

[15…7] – (R/W) – Lower Limit address (ADR_MIN) for Multiblock access

16 – 22 – (not used – read as 0)

[31…23] – (R/W) – Upper Limit address (ADR_MAX) for Multiblock access

The board that has the TOKEN will respond with data when the VME address satisfies the following condition:

$$ADR\_MIN \leq Address < ADR\_MAX.$$

Module Busy Level – Control/Status (0x0050)

[19…0] – Busy level (eight byte words)

(External RAM word count > Busy level -> module busy = 1)

[31] – Force module busy

Block Count – Control/Status (0x0054)

[19…0] – (R) - number of event BLOCKS on board

CONFIGURATION_CSR (0x0058) – (Firmware Update)

[31] – (R/W) – vme program enable

[30…28] – (R/W) – Reserved

[27] – (R/W) – Reserved

[26…24] – (R/W) – OPCODE (bit 31 = 1 also required)

[23…9] – (R) – Reserved

8 – (R) – Busy (operation in progress)

[7…0] – (R) – Last Valid Data Read

CONFIGURATION_ADR/DATA_(R/W) (0x005C) – (Firmware Update)

[31] – Execute

[30…18] – Page address

[17…8] – Byte address

[7…0] – EPROM data to write

Processor FPGA(0xd000-0xdffc)

Processor Firmware Version – Status (0xd000)

Processor CSR – Control/Status (0xd004)

0 – (R) – busy status – not used
1 – (R/W) – processor csr clear – not used
2 – (R/W) – reset (testing) N/A

Trigger source – Control/Status (0xd008)

1 – 0 (R/W) – trig setup
00 - trig on p0_trg(0) rising
01 - trig on SW TRIGGER (was internal timer)
10 - trig on internal multiplicity sum
11 - trig on p2_trg(0) rising

CTRL2 – Control/Status (0xd00C)

0– (R/W) – Enable Trigger to Module (source = Trigger source[1-0])
1– (R/W) – Enable Sync Reset to Module N/A, MOVED TO FE (0x1004)
3-2 (R/W) – Sync reset source select - default "00"
00– (R/W) – P0 Connector (VXS)
01 – (R/W) –
10 – (R/W) – VME (software generated)
11 – (R/W) – no source

Control/Status (0xd010) – used for testing

0-R/W – SW TRIGGER

BLOCK SIZE – Control/Status (0xd014)

[15…0] - (R/W) – number of events in a BLOCK.
Stored Event Count ≥ BLOCK SIZE → BLOCK CSR[2] = 1.
[31…16] – (not used)

Trigger Count – Control/Status (0xd018)

[30…0] – (R) – total trigger count
11– (R/W) – reset count

Event Count – Control/Status (0xd01C)

[23…0] – (R) – number of events on board

CLOCK_125 COUNT REGISTER (0xd020)

0 – (W) – Write '0' resets the counter. Write '1' initiates 20us counting interval.
[31 - 0] – (R) – CLK_250 counter value. (Should be 5000 after count interval.)

SYNC_IN_P0 COUNT REGISTER (0xd024)

0 – (W) – Write ‘0’ resets the counter.
[31 - 0] – (R) – SYNC_IN_P0 counter value.

TRIG2_IN_P0 COUNT REGISTER (0xd028)
0 – (W) – Write ‘0’ resets the counter.
[31 - 0] – (R) – TRIG1_IN_P0 counter value.

FE PFGA(0x1000-0xcffc)

FE Firmware Version – Status (0x1000)

FE Test Register (0x1004)
0– (R) – reset (testing) read
1– (R/W) – Collect On default on
2– (R/W) – Enable Sync Reset to Module default on, all rst sync

FE Asynchronous ADC read – Status (0xN020-0xN034) – not used

FE FIFO ADC read – Status (0xN040-0xN054) – not used

PTW (0x1058)
[8 - 0] – (R/W) – Window Width

PL (0x105C)
[15 - 0] – (R/W) – # of samples back from trigger point

PTW DAT BUF LAST ADDR (0x1060)
[11 - 0] – (R/W) – Last Address of secondary buffer (see calculation 1.0 below)

PTW MAX BUF (0x1064)
[7 - 0] – (R/W) – Max # of unprocessed PTW blocks to store in secondary buffer (see calculation 2.0 below)

NSB (0x1068)
[12 - 0] – (R/W) – # of samples before (including) trigger point to include in data processing

NSA (0x106C)
[13 - 0] – (R/W) – # of samples after trigger point to include in data processing

TET 1 (0xN070)

[11 - 0] – (R/W) – Trigger energy threshold

TET 2 (0xN074)

[11 - 0] – (R/W) – Trigger energy threshold

TET 3 (0xN078)

[11 - 0] – (R/W) – Trigger energy threshold

TET 4 (0xN07C)

[11 - 0] – (R/W) – Trigger energy threshold

TET 5 (0xN080)

[11 - 0] – (R/W) – Trigger energy threshold

TET 6 (0xN084)

[11 - 0] – (R/W) – Trigger energy threshold

CONFIG1 (0x1088)

[2 - 0] – (R/W) – Process mode select

- "000" = Select option 1
- "001" = Select option 2
- "010" = Select option 3
- "011" = Select option 4
- "111" = Run option 1 and option 4

[3] – (R/W) – Run (Collect On) default '1'

[6-5] – (R/W) – Number of pulses in Mode 1 and 2

[7] – (R/W) – Test Mode (playback)

Trigger Number (0xN08C)

[15- 0] – (R) – Trigger number

Calculation - 1.0

PTW MAX BUF = INT(2016/(PTW+8))

Where:

2016 = Number of addresses in secondary buffer

PTW = Window width in samples (Even #?)

Calculation - 2.0

PTW DATA BUF LAST ADR =

PTW MAX BUF*(PTW+8)-1

Where:

Number of bytes per trigger = PTW*125MHz

Appendix 1 – Register structure for FADC125

```
struct fa125_a24_main
{
  /* 0x0000 */ volatile UINT32 id;
  /* 0x0004 */ volatile UINT32 swapctl;
  /* 0x0008 */ volatile UINT32 version;
  /* 0x000C */ volatile UINT32 clock;
  /* 0x0010 */ volatile UINT32 pwrctl;
  /* 0x0014 */ volatile UINT32 dacctl;
  /* 0x0018 */          UINT32 blank0[(0x20-0x18)/4];
  /* 0x0020 */ volatile UINT32 serial[4];
  /* 0x0030 */ volatile UINT32 temperature[2];
  /* 0x0038 */ volatile UINT32 slot_ga;
  /* 0x003C */ volatile UINT32 a32_ba;
  /* 0x0040 */ volatile UINT32 blockCSR;
  /* 0x0044 */ volatile UINT32 ctrl1;
  /* 0x0048 */ volatile UINT32 adr32;
  /* 0x004C */ volatile UINT32 adr_mb;
  /* 0x0050 */ volatile UINT32 busy_level;
  /* 0x0054 */ volatile UINT32 block_count;
  /* 0x0058 */ volatile UINT32 configCSR;
  /* 0x005C */ volatile UINT32 configAdrData;
  /* 0x0060 */          UINT32 blank1[(0x1000-0x60)/4];
};

struct fa125_a24_fe
{
  /* 0xN000 */ volatile UINT32 version;
  /* 0xN004 */ volatile UINT32 test;
  /* 0xN008 */          UINT32 blank0[(0x20-0x08)/4];
  /* 0xN020 */ volatile UINT32 adc_async[6];
  /* 0xN038 */          UINT32 blank1[(0x40-0x38)/4];
  /* 0xN040 */ volatile UINT32 acqfifo[6];
  /* 0xN058 */ volatile UINT32 ptw;
  /* 0xN05C */ volatile UINT32 pl;
  /* 0xN060 */ volatile UINT32 ptw_last_adr;
  /* 0xN064 */ volatile UINT32 ptw_max_buf;
  /* 0xN068 */ volatile UINT32 nsb;
  /* 0xN06C */ volatile UINT32 nsa;
  /* 0xN070 */ volatile UINT32 threshold[6];
  /* 0xN088 */ volatile UINT32 config1;
  /* 0xN08C */ volatile UINT32 trig_count;
  /* 0xN090 */          UINT32 blank2[(0x1000-0x90)/4];
};


struct fa125_a24_proc
{
  /* 0xD000 */ volatile UINT32 version;
  /* 0xD004 */ volatile UINT32 csr;
  /* 0xD008 */ volatile UINT32 trigsrc;
  /* 0xD00C */ volatile UINT32 ctrl2;
  /* 0xD010 */ volatile UINT32 softtrig;
  /* 0xD014 */ volatile UINT32 blocklevel;
  /* 0xD018 */ volatile UINT32 trig_count;
  /* 0xD01C */ volatile UINT32 ev_count;
  /* 0xD020 */ volatile UINT32 clock125_count;
  /* 0xD024 */ volatile UINT32 sync_count;
  /* 0xD028 */ volatile UINT32 trig2_count;
};

struct fa125_a24
{
  /* 0x0000 */ struct fa125_a24_main main;
  /* 0x1000 */ struct fa125_a24_fe   fe[12];
  /* 0xD000 */ struct fa125_a24_proc proc;
};
```

Appendix 2 – Functions of note for FADC125

```
/****************************************************************************
*
 *
 * fa125Init - Initialize the fa125 Library
 *
 *   iFlag: 17 bit integer
 *       Low 6 bits - Specifies the default Signal distribution (clock,trigger)
 *                    sources for the board (Internal, FrontPanel, VXS, VME(Soft))
 *       bit    0:  defines Sync Reset source
 *                  0  VXS (P0)
 *                  1  VME (software)
 *       bits 2-1:  defines Trigger source
 *           (0) 0 0  VXS (P0)
 *           (1) 0 1  Internal Timer
 *           (2) 1 0  Internal Multiplicity Sum
 *           (3) 1 1  P2 Connector (Backplane)
 *       bit    3:  NOT USED WITH THIS FIRMWARE VERSION
 *       bits 5-4:  defines Clock Source
 *           (0) 0 0  P2 Connector (Backplane)
 *           (1) 0 1  VXS (P0)
 *           (2) 1 0  Internal 125MHz Clock
 *
 *     16:  Exit before board initialization (just map structure pointer)
 *          0: Initialize fa125(s)
 *          1: Skip initialization
 *
 *     17:  Use fa125AddrList instead of addr and addr_inc
 *           for VME addresses.
 *          0: Initialize with addr and addr_inc
 *          1: Use fa125AddrList
 *
 *     18:  Skip firmware check.  Useful for firmware updating.
 *             0 Perform firmware check
 *             1 Skip firmware check
 *
 */
Int fa125Init (UINT32 addr, UINT32 addr_inc, int nadc, int iFlag)
/****************************************************************************/


/************************
 *
 *  faSetProcMode - Setup ADC processing modes.
 *
 *    id    - Slot number of module to read
 *    pmode - Process mode
 *    PL    - Lookback (in samples)
 *    PTW   - Window width (in samples)
 *    NSB   - number of samples before threshold
 *    NSA   - number of samples after threshold
 *    NP    - max number of pulses
 *
 */

int
fa125SetProcMode(int id, int pmode, unsigned int PL, unsigned int PTW,
                 unsigned int NSB, unsigned int NSA, unsigned int NP)
/****************************************************************************/
```

Where:
pmode 1 = raw mode
pmode 2 = raw pulse mode
pmode 3 = pulse integral
pmode 4 = TDC

```
/*******************************************************************************
*
 *
 * fa125SetClockSource - Set the clock source
 *
 *   clksrc: defines Clock Source
 *           0  P2 Clock
 *           1  VXS (P0)
 *           2  Internal 125MHz Clock
 *
 */

int
fa125SetClockSource(int id, int clksrc)
/*******************************************************************************/


/*******************************************************************************
*
 *
 * fa125SetTriggerSource - Set the trigger source
 *
 *   trigsrc: defines Trigger Source
 *           0  P0 (VXS)
 *           1  Internal Timer
 *           2  Internal Sum
 *           3  P2
 *
 */

int
fa125SetTriggerSource(int id, int trigsrc)
/*******************************************************************************/


/*******************************************************************************
********
 *
 *  fa125ReadBlock - General Data readout routine
 *
 *    id    - Slot number of module to read
 *    data  - local memory address to place data
 *    nwrds - Max number of words to transfer
 *    rflag - Readout Flag
 *              0 - programmed I/O from the specified board
 *              1 - DMA transfer using Universe/Tempe DMA Engine
 *                    (DMA VME transfer Mode must be setup prior)
 *              2 - Multiblock DMA transfer (Multiblock must be enabled
 *                     and daisychain in place or SD being used)
 */
int
fa125ReadBlock(int id, volatile UINT32 *data, int nwrds, int rflag)
/*******************************************************************************/
```