MCRUSH Programmers Manual

Introduction

The MCRUSH is designed to receive data from 18 TDC's. The input data for the MCRUSH will be organised as can be seen in figure 1. Such data will be produced by the CSM and send over an S-LINK to the MCRUSH.

Separator	TDC0	TDC1	…	TDC17	Separator	TDC0	…

Figure 1: Input data format for the MCRUSH

A Separator word is used to mark the start of 18 time slots, one for each TDC. When there is no data available from a TDC then a NoData word is put in the data stream in the corresponding time slot.
The MCRUSH is programmed to recognise a Separator word and a TDC Trailer word. To do so, the FPGA on the MCRUSH contains two comparators. For both, Separator and TDC Trailer, a 33-bit pattern and a 33-bit mask can be programmed. The 33 bits are organised as two 32-bit words; in the last word, only bit 0 is significant. This single bit signals whether or not the first 32-bit word needs to be an S-LINK control word or not.
When a Separator is recognised a time slot counter is started which will count from 0 up until 17. While counting, data from the S-LINK input is transferred to 18 partitions (each 8K words) of the buffer memory. Partition 0 is corresponding with time slot 0, partition 1 with time slot 1 etc. The only case where no data will be transferred to the buffer memory is when a NoData word is received (NoData is defined as LD[31..0] = 0H and LCTRL_n = '1').
If there are more than 18 words send after a Separator is recognised then this data will be discarded.
When a TDC Trailer is recognised then the Event-ID of the trailer is checked whether it falls within an 8 wide window of expected Event-ID's. This can lead to an 'Accept', an 'Early', or a 'Late' condition. The Event-ID in the TDC Trailer is 12 bits wide, which means that the Event-ID can have 4096 different values. So, for 2048 values further than the expected Event-ID + 4 (which is halfway the 'Accept' window) there is a roll over from 'Late' to 'Early' (see Table 1). Note that [Expected Event-ID + 3 – 2048] is the same value as [Expected Event-ID + 3 + 2048].
'Early' and 'Late' are error conditions, which can generate an interrupt to the SHARC.
When the TDC-trailer is accepted, a flag bit is set in the 'Tetris register'. The row of the flag bit in the Tetris register is determined by the low order 3 bits of the TDC-Trailer Event-ID. The column of the flag bit in the Tetris register is determined by the TDC time slot (0..17).
When all TDC-Trailers of the Expected Event-ID are received then a 'RowOut' condition is send to the output controller.

Expected Event-ID + 3 – 2048	Early
Expected Event-ID + 4 – 2048	Late
:	:
Expected Event-ID – 1	Late
Expected Event-ID	Accept
:	:
Expected Event-ID + 7	Accept
Expected Event-ID + 8	Early
:	:
Expected Event-ID + 3 + 2048	Early
Expected Event-ID + 4 + 2048	Late

Table 1: Expected Event-ID window

Note that a row is complete when all enabled TDC's (see the TDC-Mask register) flagged the presence of their TDC-Trailer with the 'Expected' Event-ID in the buffer memory.
A RowOut condition will also occur when a row which is flagging the first, up to sixth following Event-ID is complete (Expected Event-ID + (1..6)). In this case the RowOut condition for the expected Event-ID is waiting for one or more TDC-Trailers with an Event-ID which was lost. The status of the flags of the expected Event-ID row in the Tetris Register tells us, which TDC-Trailer(s) was/were lost.
Furthermore a RowOut condition immediately occurs whenever a TDC-Trailer is received with its Event-ID equal to Expected Event-ID + 7. This will only occur if the Event-ID difference between TDC's is more than 7 which is a very rare condition. Note that in this case the last row in the Tetris register is indexed so this RowOut condition is an attempt to free up rows in the Tetris register in order to keep track of incoming TDC data.
A RowOut condition stores the data from the Tetris Register row which belongs to the current expected Event-ID, in an 'input to output FIFO' (I2O_FIFO). The expected Event-ID is also stored.
Note that the S-LINK input data is not restricted to 18 TDC number slots. This could be less since the TDC number counter is forced to 0 after each Separator word. However the frequency of the Separator words is limited due to the way the Tetris register is implemented. Separator words should at least be 3 clock cycles apart.
TDC's which are not in use or which have a malfunction can be disabled using the TDC-Mask register. When a TDC is disabled then the Tetris register will not wait for the corresponding trailer which will never come.
The output controller is supposed to read the I2O_FIFO and gather the data from the partitions in the buffer memory. This information is placed in an output FIFO (Outp_FIFO) which is read by the SHARC under DMA control with a maximum bandwidth of 80 MB/sec. Table 2 lists the format of this data.
Each word that is read from the I2O_FIFO contains the bits of the Tetris Register, and the expected Event-ID. If a bit in the Tetris Register was set then the corresponding partition in the buffer memory should be read until a TDC trailer is found with an event number

which is equal to the ‘expected Event-ID’ from the I2O_FIFO data word. It is guaranteed that the output controller will find such a TDC trailer in the buffer memory since the Tetris Register has got it’s flag set.
There can be two conditions where the output controller is switching off the read out for a certain partition. The first condition is when a buffer memory partition is full. Since the partitions are in fact circular buffers, the information in the partition is being overwritten. This means there is no unambiguous relation anymore between the content of the Tetris Register and the information in the buffer memory partition. The output controller could get confused because it is no longer guaranteed that it will find the TDC trailer with the expected Event-ID which it is looking for.
The second condition where the output controller is switching off the read out is when a single event is longer than a maximum event size. Read out of the partition is shut down to prevent this data from that partition of blocking the output bandwidth to the SHARC.

Data	Remark
MCRUSH Header	Bits 17..0 point out which TDC’s are present in the data
TDC0 Header	If TDC0 was present
TDC0 Data	If TDC0 was present and had data
:	
TDC0 Trailer	
TDC1 Header	If TDC1 was present
TDC1 Data	If TDC1 was present and had data
:	
TDC1 Trailer	
:	
:	
MCRUSH Header	Next event

Table 2: Output data format

The MCRUSH can be put in “Test Mode”. Some Control and Status registers can feed data to the S-LINK input. It is also possible for the SHARC to access the buffer Memory although this is only possible offline since the cycle-shared ports of the buffer memory are dedicated to the S-LINK input and the output controller of the FPGA during system operation.
Several interrupt sources signal special conditions or errors.
All the above can be controlled and monitored by means of a set of Control and Status Registers, which reside in the MS0 address space of the SHARC. Table 3 contains a listing these registers.

Control and Status Registers in the MS0 address space of the SHARC

Register 00H and 02H

Register 00H contains the 32-bit pattern that must match the Separator data on the S-LINK. A mask (register 02H) defines the bits that must match. If a mask bit is set to '1' the corresponding bit in the pattern register must match the Separator data on the S-LINK. If they do, a Separator is signalled.
If a mask bit is reset to '0' the corresponding bit in the pattern register may be different from the bit in the S-LINK data.
Reading back registers 00H or 02H yields the value written into them.

Address	Function	Write	Read
00H	Separator Pattern	nnnnnnnnH	nnnnnnnnH
01H	Separator Control Bit Pattern	xxxxxxxpH	0000000pH
02H	Separator Mask	nnnnnnnnH	nnnnnnnnH
03H	Separator Control Bit Mask	xxxxxxxpH	0000000pH
04H	TDC Trailer Pattern	nnxxxnnnH	nn000nnnH
05H	TDC Trailer Control Bit Pattern	xxxxxxxpH	0000000pH
06H	TDC Trailer Mask	nnxxxnnnH	nn000nnnH
07H	TDC Trailer Control Bit Mask	xxxxxxxpH	0000000pH
08H	MCRUSH Header Pattern	nnxxxxxxH	nn000000H
09H	Expected Event ID	xxxxxnnnH	00000nnnH
0AH	TDC Mask Register	xxxqnnnnH	000qnnnnH
0BH	Partition Readout Enable	xxxqnnnnH	000qnnnnH
		xxxxxxxxH	00000000H
		xxxxxxxxH	00000000H
0EH	Maximum Event Size	xxxxxnnnH	00000nnnH
0FH	Interrupt Control IRQ0	xxxxxxxqH	0000000qH
10H	LDERR_n Interrupt IRQ1	xxxxxxxxH	000nnnnnH
11H	Early and Late Event-ID; IRQ2	xxxxxxxxH	000nnnnnH
12H	Test Control Register	xxxxxxxqH	0000000qH
13H	Test Link Data Register	nnnnnnnnH	nnnnnnnnH
14H	Test Link Control Register	xxxxxxxqH	0000000qH

Remarks:
'n' is any hexadecimal number
'x' is don't care
'p' is either 00H or 01H
'q' is 00H, 01H, 02H or 03H

Table 3: Registers in the MS0 address space

Register 01H and 03H

Register 01H contains only one bit. This bit must match the control bit of the Separator data on the S-LINK but only if the mask bit (register 03H) is set to '1'. Note that the S-LINK control bit is low-active (LCTRL_n).
Reading back registers 01H or 03H yields either 00000000H or 00000001H depending on the value written into the bit 0 location. Bits 31 to 1 are always read back as '0'.

Registers 04H to 07H

The description of these registers is the same as for registers 00H to 03H, except that registers 04H to 07H control the recognition of a TDC Trailer condition. There is however a small difference between registers 00H-03H and 04H-07H. Bits 23 downto 12 in the TDC Trailer are reserved for a 12 bit Event-ID. Therefore these bits are don't care when they are written to the TDC Trailer Pattern (04H) and TDC Trailer Mask (06H) registers. When these registers are read back bits 23 downto 12 will be '0'.

Register 08H

Register 08H contains an 8 bits MCRUSH Header Pattern in the bit positions 31 downto 24. When readout is initiated a first word is send to the output stream through the Outp_FIFO, which contains these 8 bits in the positions 31 downto 24 (see table 2). The lower 18 bits of this word give information about which TDC's send data (in sequence). This is actually the AND of the Tetris Register row which is read out, and the Readout Enable register (register 0BH).
Rule: If one of the lower 18 bits is '1' then data of the corresponding TDC should be in the output stream.
Note: The following situation is an exception to this rule! If a partition full condition is met *after* the MCRUSH header is send to the output but *before* the corresponding TDC is read out and send to the output then no data is found in the output stream since the corresponding TDC was shut down before it was read out. However, this will lead to a "Partition Full Interrupt" (see register 0x0FH). This should be very rare!
Note: A Maximum Event Size overflow (see register 0x0EH) will shut down the readout as well but there should at least be some data of the corresponding TDC in the output stream!

Register 09H

Register 09H contains 12 bits which represents the event-ID that is to be expected as the next event-ID to receive from the TDC's.

Register 0AH

Register 0AH contains an 18 bits TDC-Mask. Writing a '1' to a bit enables the corresponding TDC. If the bit is written '0' then the corresponding TDC is disabled and the Tetris Register will not wait for the corresponding TDC trailer to arrive in order to generate a RowOut condition.

Register 0BH

Register 0BH contains 18 partition readout enable bits. When a bit is '1' then the corresponding buffer memory partition is enabled for readout, if it is '0' then the partition is skipped during readout. Bit 0 corresponds with TDC0; Bit 1 with TDC1 and so on.
There are two conditions where readout for a TDC is disabled by the hardware.
The first condition is when the corresponding TDC partition in the buffer memory is full. This occurs when more than 5 to 6K words are stored in the buffer memory which are waiting for readout. If this is the case then it is no longer possible to have a consistent image of the TDC event data in the buffer memory with respect to the content of the Tetris Register. Therefore readout may not be able to find the TDC Trailers anymore for which it is looking in the buffer memory. To avoid readout problems the readout is disabled. This condition should never occur since data should be read from the output FIFO fast enough.
Note that the Partition Full condition is met when a partition contains 5 to 6K words in stead of 8K words. This is due to the fact that the FPGA contains a pipeline to the buffer memory and only the upper 3 bits of the read- and write pointers of the partitions are used (to save hardware resources in the FPGA) to determine a full condition. The condition is met when:

```
WritePointer[12..10] = ReadPointer[12..10] – 2
```

Note the following example: when the read pointer is 0x0423 a partition full condition is met when the write pointer increases to 0x1C00, a difference less then 6K words.
The second condition where readout for a TDC is disabled is when during readout of a single event a maximum event size is exceeded. In such a case it is likely that a TDC is out of order. Such a condition forces the readout for that particular TDC to be disabled. This prevents lots of wrong data from blocking the output bandwidth to the SHARC
In either case IRQ0 is asserted (if enabled, see register 0FH Interrupt Control IRQ0)
The SHARC can read the TDC readout enable register (see register 0BH). This gives the SHARC information about which condition and TDC caused the interrupt. The SHARC can write to the register as well, but enabling readout should only be done during initialisation of the MCRUSH. If the readout is enabled during runtime then the behaviour depends on the values op the read- and write-pointers of the buffer memory partition. Therefore this could result in an immediate assertion of the partition full interrupt which shuts down the output again. It could also result in a maximum event length interrupt since the relation between the buffer memory partition data and the Tetris register is lost so the hardware may keep searching for a TDC trailer which was overwritten.

Registers 0EH

The 12-bit value in this register determines the maximum allowable event size for a TDC. If the number of words in an event which is being read out from a partition in the buffer memory is more than the value in this register then the event is truncated. Readout for the partition is stopped by clearing the corresponding partition readout enable bit in register 0BH. A “Readout Maximum” interrupt will be generated, to signal the SHARC that something went wrong (see Register 0FH).
Register 0EH will power up to a default value of 1024.
Note that the value ‘0’ for this register will give unpredictable results and should be avoided.
Note also that due to pipelining, one extra word could be transferred to the output FIFO.

Register 0FH

This register controls the interrupt output IRQ0_n to the SHARC.
There are three interrupt sources. If register 0FH is read back then bits 0 to 2 determine which of the three interrupt sources generated the interrupt.
Bit 0: I2O_FIFO Full Interrupt.
Bit 1: Buffer Memory Partition Full Interrupt.
Bit 2: Readout Maximum Interrupt.
Writing a ‘1’ to one of these bits in register 0FH clears the interrupt.
Bits 3 to 5 are the mask bits for the interrupt bits (bits 0 to 2). An interrupt bit can only be set when the corresponding mask bit is a ‘1’.

Bit 3: I2O_FIFO Full Interrupt Mask. If ‘1’ then an I2O_FIFO Full condition generates an interrupt.
Bit 4: Buffer Memory Partition Full Interrupt Mask. If ‘1’ then a Buffer Memory Partition Full condition generates an interrupt. Note that the Partition Readout Enable Register (Register 0BH) can give more detailed information about which partition caused the interrupt.
Bit 5: Readout Maximum Interrupt Mask. If ‘1’ then an interrupt is generated if an Event is read from a partition but the event contains more data words than is programmed in the “Maximum Event Size” register 0EH. Note that the Partition Readout Enable Register (Register 0BH) can give more detailed information about which partition caused the interrupt.

Register 10H

This register controls the interrupt output IRQ1_n to the SHARC.
IRQ1_n occurs whenever an error is detected on the S-LINK interface.
When reading this register, bit 18 of this register indicates whether the link error occurred during the transfer of a TDC data word (Bit 18 = ‘0’) or whether the link error occurred during the transfer of another word (Separator, NoData or other data word etc.).

If bit 18 = ‘0’ then bits 17 downto 0 yields a buffer memory address. On this buffer memory address, the TDC data word is stored which contains an error indicated by the LDERR_n bit of the S-LINK.
If bit 19 of this register is a ‘1’ then there was an overrun condition. This means that there were one or more errors that weren’t read out in time. Note that during an overrun condition, bits 18 downto 0 of the register represent the status of the first error that occurred.
Writing any value to this register clears IRQ1_n.

Register 11H

This register controls the interrupt output IRQ2_n to the SHARC.
IRQ2_n occurs whenever a TDC trailer is received with an Event-ID which is out of the expected Event-ID window. Such an Event-ID can be either ‘Early’ or ‘Late’ (see table 1).
Bits 11 downto 0 represent the Event-ID which caused the interrupt.
Bits 16 downto 12 represent the TDC slot number which caused the interrupt.
If Bit 17 is ‘1’ then the Event-ID is ‘Early’. For example, this will be the case when the expected Event-Id is 51 and the received Event-ID for a TDC is 102, which is ‘in the future’ with respect to the expected Event-ID.
If Bit 18 is ‘1’ then the Event-ID is ‘Late’. For example, this will be the case when the expected Event-Id is 51 and the received Event-ID for a TDC is 50, which is ‘in the past’ with respect to the expected Event-ID.
If bit 19 of this register is a ‘1’ then there was an overrun condition. This means that there were one or more errors that weren’t read out in time. Note that during an overrun condition, bits 18 downto 0 of the register represent the status of the first error that occurred.
Writing any value to this register clears IRQ2_n.
Take care in enabling IRQ2 on the SHARC. When the MCRUSH is being synchronised with respect to the Event-ID’s, the expected Event-ID register (register 09H) is programmed with an Event-ID, which will be valid in the near future. ‘Late’ interrupts will be generated until the Event-ID’s form the TDC trailers are within the expected Event-ID window. Therefore it is advisable to enable IRQ2, only when the MCRUSH is in synchronisation and Event-ID’s should fall within the expected Event-ID window.

Registers 12H to 14H

These registers are used in ‘Test Mode’.
In test mode, the S-LINK input is disabled, and is taken over by the SHARC. The SHARC can now write data as if it came from the S-LINK.
The Test Control Register (12H) controls access to the buffer memory as well. The buffer memory has two cycle-share ports.
The first port is always dedicated to the S-LINK input.
The second port is connected to the FPGA by default in such a way that the FPGA can read data from the buffer memory. For test purposes the SHARC may read or write the

buffer memory as well. When bit 2 of the Test Control Register is set then the second port is switched from ‘FPGA Read mode’ to SHARC ‘Read/Write mode’.

Register 12H is the test control register. This register contains the following bits:
Bit 0: If in test mode, taking this bit from 0 to 1 writes test data from register 13H and 14H to the MCRUSH S-LINK input.
Bit 1: ‘0’ Normal operation (S-LINK)
‘1’ test mode operation.
Bit 2: ‘0’ Normal operation. FPGA read mode; default on power up.
‘1’ test mode operation. SHARC has got Read/Write access to buffer memory.

Register 13H are the 32 data bits which would normally come from the S-LINK LD0-LD31 bus.

Register 14H contains the following bits:
Bit 0: Status of the LCTRL_n bit which would normally come from the S-LINK.
Bit 1: Status of the LDERR_n bit which would normally come from the S-LINK.

After power-up, the MCRUSH operates normal (that is, no test mode and the FPGA has got read access to the buffer memory).
To run a test, write the following data in the corresponding registers:
12H: 00000002H (Enter the test mode)
13H: S-LINK Data word 1
14H: S-LINK Corresponding status LCTRL and LDERR
12H: 00000003H (Write Data Word 1 by taking bit 0 from 0 to 1)
12H: 00000002H
13H: S-LINK Data word 2
14H: S-LINK Corresponding status LCTRL and LDERR
12H: 00000003H (Write Data Word 2 by taking bit 0 from 0 to 1)
12H: 00000002H
:
13H: S-LINK Data word N
14H: S-LINK Corresponding status LCTRL and LDERR
12H: 00000003H (Write Data Word N by taking bit 0 from 0 to 1)
12H: 00000000H (Go to normal operation, exit test mode)

SHARC Memory regions:

Memory Region	Function	EBxWS	EBxWM	Remarks
MS0	MCRUSH internal registers	100	01	1, 4
MS1	Read Output FIFO	000	00	1, 2
MS2	Buffer memory access when in SHARC Read/Write Mode	011	01	1, 3
MS3	Not Used	N.A.	N.A.	

Table 4: SHARC Memory regions

Remarks:
1: EBxWS and EBxWM are sub-patterns of the Wait register described in chapter 5.4.4.1 of the SHARC user's manual.
2: EBxWS = 000 means 0 Wait, 0 Hold Cycles
EBxWM = 01 Internal Wait states only
3: EBxWS = 011 means 3 Wait, 0 Hold Cycles
EBxWM = 01 Internal Wait states only
4: EBxWS = 100 means 4 Wait, 1 Hold Cycle
EBxWM = 01 Internal Wait states only

SHARC Flags:

Table 5 gives an overview of the SHARC flags.
Flag 0 and Flag 1 will be discussed in the section ‘Reset Facilities’ below.
Flag 2 is connected to the Freeze_n signal. When asserted, this signal stops the data flowing from the S-LINK FIFO into the FPGA. This eventually will cause an UXOFF# condition on the S-LINK when the S-LINK FIFO fills up to ‘half full’. The output controller is not affected by the freeze_n signal. If Flag 2 is ‘0’ then the Red LED is on.
Flag 3 must be configured as an input and is connected to the Empty output of the Output FIFO.

Flag Number	Direction	Description
0	Output	S-LINK Reset
1	Output	FPGA Reset
2	Output	Freeze/ Red LED
3	Input	Empty status of Output FIFO

Table 5: SHARC Flags

SHARC Interrupt Lines

IRQ0 is asserted when the I2O_FIFO is full, a buffer memory partition is full or an event was read out which was longer than a pre-programmed maximum (see register 0FH).
IRQ1 is asserted when an S-LINK error (LDERR_n) occurs (see register 10H).
IRQ2 is asserted whenever a TDC trailer is received with an Event-ID which is out of the expected Event-ID window (‘Early’ or ‘Late’, see register 11H and table 1).

Reset Facilities

There are three reset signals: “GlobalRst_n”, a “Rst_n” and “LRst_n”.
The GlobalRst_n signal becomes active during a power-up of the MCRUSH, during a “Manual Reset” (reset button), or an “External Reset” (2-pin header). The GlobalRst_n signal resets the SHARC and the FPGA (including the Pipeline, Address Generators, Tetris Register, I2O_FIFO, Output FIFO etc.). Note that the S-LINK and the S-LINK FIFO are not reset.
The “LRst_n” signal resets the S-LINK and the S-LINK FIFO. The S-LINK will force LDOWN# low until the initialisation phase of the S-LINK is complete. LDOWN# will then go high again. The S-LINK is now up and running again. “LRst_n” can be controlled by the SHARC using flag0.
The “Rst_n” signal is used to reset the whole FPGA (including the Pipeline, Address Generators, Tetris Register, I2O_FIFO, Output FIFO etc.). “Rst_n” is active during a GlobalRst_n, and can be controlled by the SHARC using flag1.
The following sequence should give a proper initialisation. First there will be a global reset. After that, the SHARC is booted through one of its links. In the mean time garbage data had a chance to enter the S-LINK FIFO and the FPGA. Therefore the SHARC activates flags 1 (Rst_n) and flag 0 (LRst_n). This resets the S-LINK, the S-LINK FIFO and the FPGA.