Draft, Version 5.0
Apr 16, 1997

Interfaces to the SVT Hit Finder Board

J. Berryhill, R. Culbertson, H. Frisch, M. Shochet, P. Wilson
with the SVT and SVX-II Groups

1 Introduction

In this draft note we begin to specify the hit finder board of the SVT. After a L1 accept, this board receives the SVX data from the silicon readout system over fiber optic G-links. It forms clusters of hit strips and computes the centroids of the clusters. Through a front panel connector, the hit finder passes the cluster information to a merger which passes it on to the associative memory sequencer. In this note we specify the input and output interfaces of the hit finder.

This is a work in progress, not all information is final. There are also some questions inserted where the information is not available yet. I have borrowed from the references.

1.1 Overview of the SVX

The SVX consists of 3 physical barrels in z, each consisting of 2 logical barrels. In this note, barrel refers to a logical barrel, numbered 0 to 5 from $-z$ to $+z$. Each barrel consists of 12 wedges in ϕ, 0-11, numbered from $\phi = 0$, increasing with increasing ϕ. In this note, "wedge" refers to one ϕ segment of one barrel. Each wedge has 5 layers, numbered 0-4, from small radius to large. Unless otherwise noted, layer will refer to a layer in a single wedge and barrel.

2 Overview of the SVX Readout

Each layer is read out out with the SVX-III chip with 128 strips handled by one chip. The chips suppress ADC's below a programmable threshold. There is one threshold value for a chip and that threshold applies to all strips on that chip. The chips may be programmed to digitize with 7 or 8 bits; the 7-bit digitization takes approximately 1.25μs. Table 1 summarizes the SVX layers and their chips.

A set of readout chips is mounted on a hybrid card which is mounted on the silicon wafers. There is one hybrid for each layer and it reads out both ϕ and z. Each hybrid is

Layer	z chips(channels)	ϕ chips(channels)
0	2(256)	2(256)
1	3(384)	3(384)
2	5(640)Stereo	5(640)
3	4(512)	6(768)
4	7(896)Stereo	7(896)

Table 1: SVX channel and readout chip counts.

read out along a High-Density Interconnect cable (HDI). An HDI transports 8 parallel bits from a single hybrid at 53Mhz. This is often called a stream or bus.

The ϕ chips are read out before the z or stereo chips. The ϕ strips are read out in the order that they physically appear on the layer. The readout order may be in either the positive ϕ direction or the negative ϕ direction. However, all layers in the same wedge will be read out in the same direction. Note that we expect that for the 6 barrels at the same ϕ, barrels 0, 2, and 4 will be read out in one direction and barrels 1, 3, and 5 will be read out in the other direction. This is because the first set of barrels are read out from one end and the second set are read out from the other end of the physical barrel.

The hit finder outputs hits corresponding to the readout direction. For wedges read out in the positive ϕ direction, the strips with the smallest ϕ will have the smallest strip number. For wedges read out in the negative ϕ direction, the strips with the *largest* ϕ will have the *smallest* strip number. All information necessary to translate the strip number to a physical location will be contained in the associative memory roads. This includes reversing the layers with $-\phi$ readout and correcting for misalignments.

Each HDI terminates on a port card (PC) mounted on the SVX support structure. There is one PC for each wedge so it receives data from 5 streams, one for each layer. On the PC each 8-bit stream is converted to 8 parallel fiber optic lines, also running at 53MHz. The conversion is done by a Dense Optical Interface Module (DOIM).

The DOIM optical signals run to Fiber Interface Boards (FIB's) [1] in a VME crate in the pit. The FIB will do channel-by-channel pedestal subtraction on the data. It also inserts the stream ID, bunch number and end-of-event records in the data. Lastly, the FIB combines sets of HDI data streams and multiplexes them on one G-link, a 1.5GHz serial fiber optic system.

Which streams are combined on a G-link is somewhat flexible. All 5 streams from a PC must go to the same FIB due to the design of control signals. How the data from those 5 streams is split among the G-links is determined by the FIB aux card design. FIB's have 4 G-links each and handle 2 wedges (10 data streams) as explained more below.

The G-link fiber optic lines ($50\mu/125\mu$ multimode with ST connectors) run upstairs to the first floor. At this point each fiber-optic signal is divided into two identical fiber-optic signals by a splitter box to be designed. One set of outputs is terminated by G-link receivers on VME Readout Boards (VRB), located near the splitter box on the first floor. The VRB system is 36 boards (4 G-links per board) divided among 6 VME crates.

On the VRB board, the data are stored in one of 4 L2 buffers to wait for a L2 decision. There are control lines running from each VRB back down to its corresponding

FIB. (There is a one-to-one correspondence between VRB's and FIB's.) One of the VRB crates will contain a Silicon Readout Controller (SRC) board which manages the VRB boards. The SRC controls are fanned out to the other VRB crates.

The second set of G-link outputs from the splitter box goes to the SVT hit finders on the second floor. Because the data are optically split, the two streams are identical and the z data will be sent to the hit finders even though they do not use it.

The SVT G-link receivers will be mounted on transition cards, four per card [2]. The Finisar optical receiver and the Hewlett-Packard G-link chip set serial-to-parallel converter will be mounted directly on the cards. The SVT and the VRB can use the same card.

3 Hit Finder Input

Figure 1 shows the readout organization.

3.1 Data format

Table 2 displays the organization of the data entering the hit finders. Each data stream begins with four 8-bit ID bytes. The first two bytes identify which HDI sent this data. The values are programmable. The third ID byte will be the bunch crossing number, counted from a bunch 0 crossing. The fourth byte (RDQ) will be a flag indicating the backend SVX-III chip state while the frontend chip was storing the charge for this event. (The pedestals may affected by the backend state.) The byte values are: 0, quiescent; 1, digitize mode; 2, read mode. These bytes are inserted in the data stream by the FIB.

Following the stream ID bytes is the programmable 8-bit chip ID of the first chip read out in the stream, then an 8-bit status byte from that chip. This status byte includes the pipeline capacitor number (PCN) where this data originated. The SRC knows which PCN is expected and checks that all chips are reading out the correct capacitor. Following that is the data from the chip: 8-bit chip channel number (0-127) and 8-bit ADC data. Following that is another chip ID, status, and data until all chips in the stream are transmitted. If a chip has no data, its chip ID and status bytes are still transmitted and no data follows those words. The end of the data stream will be indicated by four end-of-record (EOR) bytes.

A chip ID byte can be recognized by always having its MSB turned on and the next bit turned off: 10xxxxxx. This pattern distinguishes a chip ID from an ADC channel number since the MSB of an ADC channel is never on. There will be only 4 to 14 (2 to 7 ϕ) chips contributing to a data stream so there are some spare bits in the chip ID byte. To easily separate the readout of ϕ chips and z chips (which the hit finder ignores), bit 5 asserted indicates a chip ID for a ϕ readout chip. This leaves the 5 LSB bits, (32 values) for actually identifying the ϕ or z chip. This is summarized in Table 3 where x implies a variable bit.

The chip ID's are programmed such that the first chip to be readout will have the 4 LSB of chip ID set to 0000, the next 0001, etc. This pattern allows the 4 LSB of the chip ID to be passed along as the MSB of the output strip number.

EOR will be 11xxxxxx which signals the end of data on this stream for this event. There are at least three ways to end the readout which are reported in the LSB of the EOR byte. The first is the normal termination where all the channels over threshold are transmitted. This is indicated by EOR_0. The FIB will have the capability of truncating

ID_0
ID_1
ID_2
ID_3
chip ID
status
Channel #
data
Channel #
data
$\vdots$
chip ID
status
Channel #
data
$\vdots$
Channel #
data
EOR_n
EOR_n
EOR_2
EOR_2

Table 2: The format of the 8-bit SVX data stream.

the transmission of a data stream if it is longer than a preset limit. This is called overflow truncation and is flagged by EOR_1. The SRC also has the capability of aborting the readout, this is indicated by EOR_3. Following the two EOR's which indicate the status of the readout will be at least two EOR_2's. EOR_2 is called "Fill EOR".

As explained more below, there is more than one data stream on a G-link. Since these streams are of variable length, one can end before the others. If one stream finishes before another, the shorter stream will contain EOR_2's while the longer stream is finishing sending data.

While any of the streams on a G-link contain data, the G-link's data valid line (DAV) will be asserted. After the longest stream sends its EOR's, DAV will be deasserted. Since the SVX-III chip cannot digitize while it is reading out, DAV is always deasserted for $> 1.25\mu s$ between events. Because the FIB packs the data, there is no chance of the continuous transmission of data being interrupted, even if there is a pause in the SVX-III chip readout. This means DAV will be continuously asserted from the start of the event to the end. Since the first stream ID byte may contain any bit pattern, it can't be used to indicate a new event. Since the number of EOR_2's is variable, the new event can't be found as the first data after some number of EOR's. There is only one way to identify the beginning of a new event: the first data after the long ($> 1\mu s$) deassertion of DAV during

bit	7	6	5	4	3	2	1	0
channel #	0	x	x	x	x	x	x	x
ϕ chip ID	1	0	1	x	x	x	x	x
z chip ID	1	0	0	x	x	x	x	x
EOR	1	1	x	x	x	x	x	x
EOR_0 (EOR_n)	1	1	0	0	0	0	0	1
EOR_1 (EOR_t)	1	1	0	0	0	0	1	0
EOR_2 (EOR_f)	1	1	0	0	0	1	0	0
EOR_3 (EOR_a)	1	1	0	0	1	0	0	0

Table 3: Flag bit patterns in the SVX data stream.

digitization.

3.2 FIB-SVT Organization

Each hit finder board receives signals from six G-link lines running in 20-bit mode. The receivers will be mounted on a transition card and the data are transmitted through the P3 connector to the main board where clusters are formed. Each 20-bit stream consists of an 8-bit stream from one channel, another 8-bit stream from a second channel and half of a third stream (4bits). The split stream will have LSB on the lower numbered G-link and MSB 4 bits on the higher numbered G-link.

Table 4 lists the inputs to the hit finders for 1/12 of the SVT system. All the data received by one hit finder board must be from the same ϕ sector. One hit finder handles all layers from 1 wedge and 2 barrels. Therefore 1/12 of the system consists of 3 hit finders which handle all layers from 1 wedge of 6 barrels. The total number of hit finders is 36.

At this point we should note a potential problem. In the design of reference [3], all the wedges fed to a single hit finder board are from the same ϕ, as described above. The G-links from the FIB will come in ribbons of fibers and the streams grouped on each ribbon are from the same barrel and different ϕ's. For example, with a 12 G-link ribbon, the first ribbon contains the data from barrel 0 and ϕ 0-5. The ribbon would have to be split and sent to 3 different crates if the crate layout of reference [3] is used. With 144 total G-links and 2 per wedge, there are 72 individual fiber optic routings between 6 crates.

One solution is to run only pairs of fibers from the splitter to the SVT. In this plan there are no fiber ribbon to individual fiber fanouts on the SVT and the cable routings will be easily hidden in a cable tray.

3.3 G-link Lock

Lock refers to the G-link transmitter and receiver being synchronized. The G-link maintains this synchronization with 4 control bits sent between every 20-bit word. There is a basic sync called frequency lock and a secondary sync called phase lock. There are two signals transmitted from the VRB G-link receivers to the FIB G-link transmitters via command words from the SRC. The first indicates the receiver is in frequency lock; the second indicates it is in phase lock. A system with no lock is state 00, with frequency lock is state 01, and

Barrel	Layer	z chips	ϕ chips	G-link	Hit Finder
	0	2	2	0L	
	1	3	3	0M	
0	2	5St	5	1L	
	3	4	6	1M	
	4	7St	7	0H,1H	0
	0	2	2	2L	
	1	3	3	2M	
1	2	5St	5	3L	
	3	4	6	3M	
	4	7St	7	2H,3H	
	0	2	2	4L	
	1	3	3	4M	
2	2	5St	5	5L	
	3	4	6	5M	
	4	7St	7	4H,5H	1
	0	2	2	0L	
	1	3	3	0M	
3	2	5St	5	1L	
	3	4	6	1M	
	4	7St	7	0H,1H	
	0	2	2	2L	
	1	3	3	2M	
4	2	5St	5	3L	
	3	4	6	3M	
	4	7St	7	2H,3H	2
	0	2	2	4L	
	1	3	3	4M	
5	2	5St	5	5L	
	3	4	6	5M	
	4	7St	7	4H,5H	

Table 4: The hit finder input organization of all barrels at one ϕ in the SVX (1/12 system). St=stereo. L indicates low order 8 bits in G-link output, M indicates middle 8 bits, H indicates high 4 bits. When a stream is split between two G-links, the low-order bits go on the lower-numbered G-link and the high-order bits go on the higher-numbered G-link.

with complete lock is state 11. There is no 10 state, when lock is lost, the state becomes 00.

When an FIB sees its sorresponding VRB receivers in state 00, it sends fill frames of the type 0 (FF0) to regain frequency lock. When the receiver has regained frequency lock, it communicates state 01, which causes the the transmitter to send fill frames of type 1 (FF1). When the receiver regains phase lock, it communicates state 11, and the system is available for data. With direct feedback this process requires on the order of 3ms.

Here are some principles guiding the design of the lost-lock error handling. Lock needs to be re-established anytime any of the G-link connected systems (FIB/VRB/SVT) is powered up. Lost lock is expected to happen very rarely, usually caused by a serious event which would effect all trigger components, such as a power glitch. If lock is lost during data taking, the trigger should be stopped by signaling a fatal error to the TSI. The data in the trigger at this point are corrupted and not easily recovered so the trigger needs to be flushed and restarted. Which G-link lost lock must be recorded, so that the errors can be monitored. There must be the capability of masking off any particular G-link such that it cannot report lost lock. This is for the case when a link is known to be bad but data taking must continue.

Although the power-up lost-lock state and the error lost-lock state are different, it would be logical to handle the two cases similarly if not identically.

Since the trigger must be flushed and restarted on a lost-lock, the FIB/VRB/SVT system cannot handle the error alone. The current plan is for the TSI to handle a lost lock in conjunction with the VRB and SVT. For a complete description of the TSI and Read Out Controller (ROC) communication see ref [12] and [13].

Briefly, a ROC is whatever handles the readout for the crate, in our case, the TRACER board in all the VRB and SVT crates. All ROC's can indicate an error condition to the TSI over a dedicated line. When it receives the error, the TSI will stop the trigger by sending a HALT message to all crates. At this point the TSI may initiate VME action to identify the error. Shortly after the HALT, the TSI issues a RESET message and all boards should deassert the error line and begin resetting. After a preset length of time (longer than 30ms) the TSI assumes all boards are reset and issues a RUN command. Data taking resumes at the next bunch 0 crossing. If an error condition still exists, the error line is asserted again and the process is repeated.

The current plan is for the SVT or the VRB to signal an ERROR to the TSI when lock is lost. This will trigger the HALT/RESET/RUN cycle. The VRB will know if it has caused the ERROR cycle due to lost lock, but it does not know if the SVT caused it due to lost lock on one of its boards. Since it doesn't know, the VRB will go through the lock recovery procedure every time the ERROR cycle is triggered. Furthermore, there is no information on the state of the SVT receivers during this process so the VRB can't do the usual recovery using the two state feedback lines. This means the fill frames would have to sent for a generous preset time period, for example, FF0 for 10ms and FF1 for 10ms. If lock had not been reestablished during this time, a second error would occur immediatly after the RUN command.

Which link failed would be stored on the board or in the crate controller until a general VME interrogation of all trigger systems is done at some later time. This system has the advantage that it is probably the fastest way to recover, which would be important if lost lock ever became a frequent occurrence. In this way, building speed into the recovery is conservative.

If we are comfortable with the assumption that lost lock is expected to be very rare while taking data, then it is insignificant how long it takes to reestablish lock. A second, VME based scheme would start the same way: the TSI stops the trigger after it receives the error. At this point, the VMEbus CPU (essentially run control) takes over. It does the interrogation of the trigger for the error, determines the cause was lost lock and who was the culprit, and then proceeds to reestablish lock:

- VMEbus CPU instructs all FIB's to send FF0.
- VMEbus CPU waits for all VRB/SVT boards to report state 01.
- VMEbus CPU instructs all FIB's to send FF1.
- VMEbus CPU waits for all VRB/SVT boards to report state 11.
- VMEbus CPU instructs all FIB's that lock is achieved.

This feedback procedure could also be run as a part of the establishment of the initial lock before the TSI is enabled by run control (or even built).

The fact that there is another CPU in the feedback loop is probably an advantage for flexibility, and a disadvantage in terms of complexity and interdependence of systems.

A third, backup strawman scheme might also be implemented with few or no modifications. In this plan the stream which lost lock would simply be turned off. The receiver boards would generate a replacement data stream with all empty events. These would be flagged in the data stream as errors.

Since the first scheme is the current preferred scheme and the second two may be simple to implement, it's possible that all three will be available.

3.4 Other Failure Modes

There are many ways for the SVT input to fail. The size of the data stream may be longer than is physically possible. In this case, there are two possibilities. The hit finders would continue working, outputting a long nonsense event. The SVT would never signal end-of-event to L2 so L2 could never finish. L2 would start causing deadtime and eventually time out.

The second possibility is that unphysically long event fills up the SVT such that the hit finder output can no longer be accepted by the second stage of the SVT. In this case, the hit finder's buffers would overflow. Since the buffers will be big enough to hold 100% occupancy for four events, an overflow should never occur and is a fatal error which should be signaled to the TSI. In this case it may also be possible to signal a non-fatal error and keep running by dropping some data, and passing only EOR's through.

Another possible failure mode would be no data. The links continue running but no data shows up. This would also result in no end-of-event being sent to L2 and L2 eventually timing out. If any of these errors occur on all links at once or just a single link, the result is the same.

It is possible that the EOR will occasionally be lost. The stream that lost it would continue to wait until the EOR of the next event. In the meantime the SVT will not send an end-of-event to L2 and L2 would again be creating deadtime. This would occur on every subsequent event so it would be very noticable. This error could be identified in other ways such as seeing data in the order ϕ, z, ϕ or seeing the 1μs digitization gap before seeing the EOR.

3.5 Occupancy

For estimates of the occupancy see [5], [6], and [7].

4 Hit Finder Output

The output of the hit finder is a single stream of clusters on a front panel connector. This information is sent to a front panel connector of a merger board. The output of this merger then goes to an associative memory sequencer.

4.1 Data Format

Since all the inputs to a single hit finder board are from the same ϕ, no ϕ information is output. Since the input to a single hit finder involves five layers and 2 or 3 barrels, layer and z information is output.

A 14-bit cluster centroid is composed of two parts. The most significant 10 bits are the strip number of the center strip from the input. This is actually computed from the chip number and the channel number within that chip: (chip number)*128+channel number. The 4 least significant bits indicate the result of combining up to three strips into a cluster and charge weighting them to find the centroid. One bit is used to indicate the cluster was found as part of a long cluster, 4 or more strips.

There are 3 bits for the layer and 3 bits for the barrel. In reference [3], the numbering for barrels goes from 1 to 6 and the numbering for layers is 1 to 5, layer 0 being reserved to indicate an XFT track.

Table 5 is an example data stream. The control bits are defined in the next subsection.

bit	4	3	2	1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6	5	4	3	2	1	0
use	AF	DS	EE	EP	data																				
clus				EP	layer			z			LC	centroid													
clus				EP	layer			z			LC	centroid													
											⋮														
clus				EP	layer			z			LC	centroid													
clus				EP	layer			z			LC	centroid													
end			EE	EP	error flags												PA	event tag							

Table 5: An example data stream of the hit finder to merger interface. AF=Almost Full, DS=Data Strobe, EE=End of Event, EP=End of Packet, PA=PArity, LC=Long Cluster.

The event tag field is the 8-bit bunch counter transferred from the input stream. The error bits are not yet defined. In the end event word bit 8 is reserved for a parity bit. This bit is the bitwise XOR of the low order 23 bits of all words in the data stream for one event, excluding the end event word.

4.2 Output Protocol

The protocol is the standard SVT internal protocol [8], which is the official document. We briefly summarize it here.

The transfer is by 25 bits (50 pins) of LVDS. The source initiates the transfer of data, each data word is indicated by bit 23, the clock or data strobe (DS). The receiver module will be filling a FIFO with the data.

Bit 24 is almost full (AF) and is set high by the receiving board when its FIFO is almost full. When this bit is asserted, the source stops the transfer until the bit is cleared. Since almost full is signaled significantly before the receiver is actually full, the source is allowed many clock cycles to respond. Bit 21 is End Packet (EP) and is used to indicate the end of a logical group of data words. Bit 22 is called End Event (EE) and indicates that this is the last word being sent for this event. An EE must be sent for every event, whether or not and data words were sent.

pin	use	pin	use
1	centroid lsb_	2	centroid lsb
3	centroid_	4	centroid
	⋮		
29	centroid msb_	30	centroid msb
31	barrel lsb_	32	barrel lsb
33	barrel_	34	barrel
35	barrel msb_	36	barrel msb
37	layer lsb_	38	layer lsb
39	layer_	40	layer
41	layer msb_	42	layer msb
43	EP_	44	EP
45	EE_	46	EE
47	DS_	48	DS
49	AF_	50	AF

Table 6: Pinout for the hit finder to merger interface.

4.3 Output Pinout

4.4 A Note on the Intermediate Silicon Layers

The ISL is one additional layer of silicon detectors around SVX-II and an additional 2 layers between the SVX-II and the plug. They are all double sided with $r - \phi$ readout on one side and stereo on the other. The ISL is not expected to be used in the SVT trigger but it may in the future so the hit finder should be designed with this in mind, if possible.

There are two factors directly effecting the hit finder. SVX-II has at most 7 ϕ chips in a stream but the ISL will have 8 ϕ chips in every stream. This means that the largest chip number has become valid data and can't be used for a flag value and can't be compacted away.

The second is the number of bits needed on the output stream to uniquely identify the cluster to the AM system. Recall that pairs of fibers that share a data stream must be kept on the same hit finder. In SVX-II the pair is contained in a 30° wedge. In the plug ISL, the pair would span two 30° wedges for one barrel, including one stream that crosses the wedge boundary. See reference [14] for details. In the central ISL the pair represents 5 30° wedges in one barrel.

The most likely use of the ISL in a trigger is to make tracks from the plug ISL and 4 layers of SVX-II. There are many choices of how a system could be arranged. The actual decisions would have to include informaiton on occupancy, geometry, AM size, and the gaols of the trigger. Here we propose one just based on DAQ organization.

Suppose 2 30° wedges are combined in one AM system and there are different AM systems for the East and West plugs. One hit finder could handle the data from the 2 wedges and both barrels on one end of the ISL (10 streams). We allow for 3 bits for the layer number so it would be no problem to use the correct layer number. The ϕ segmentation now has to be continuous along 6144 strips instead of the 896 maximum in the SVX-II, requiring 13 bits instead of 10. The z segmentation is only 1 bit. If the long bit is absorbed into the ϕ field, then there are enough bits to maintain the cluster resolution and have the cluster fully specified in one word. The hit finder usually simply appends layer and z bits to each stream. In this scenario it would have to also overwrite the long bit and actually do an addition to find the ϕ field. The easiest way to handle this would be to run the data through a programmable chip between the individual streams and the output.

This ISL data would then have to be merged with the SVX-II data from the same two wedges and the nearest two barrels (two SVT hit finders). Since there are no extra bits in this data stream, adding the extra ϕ bit would require modifications. The cleanest solution would be to have two spigots on the hit finder, each output would be run through a programmable chip to deliver the correct encoding. This also answers the question of where the data stream would be split.

A second way to handle the encoding is to modify the data stream in a special merger board as it is combined with the ISL data. How the SVX-II data is split to the ISL trigger merger board is not specified.

References

[1] Kerry Woodbury *Fiber Interface Board* ESE-SVX-951010, Feburary 11, 1997.

[2] John Dusatko *The G-link Fiber Receiver Transition (GRT) Board* September 4, 1996. (To be updated.)

[3] SVT group, *SVT Technical Design Report.*

[4] VRB group, *SAR to SVT Processor Interface Protocol* (draft).

[5] Hsaio Ying Chao and Jaroslav Antos *Occupancy Study for the SVXII* CDF note 2386, Nov 21, 1994.

[6] Ray Culbertson *Occupancy in SVXII* December 12, 1995.

[7] P. Murat *The New Plug and Occupancies in Run II CDF Silicon Tracker* CDF note 3720, March, 1997.

[8] S. Belforte *et al Specification of the XTRP, SVT, and Level 2 Interfaces* October 15, 1996.

[9] SVT group, *SVT I/O Data Format Specifications*, internal note 54.

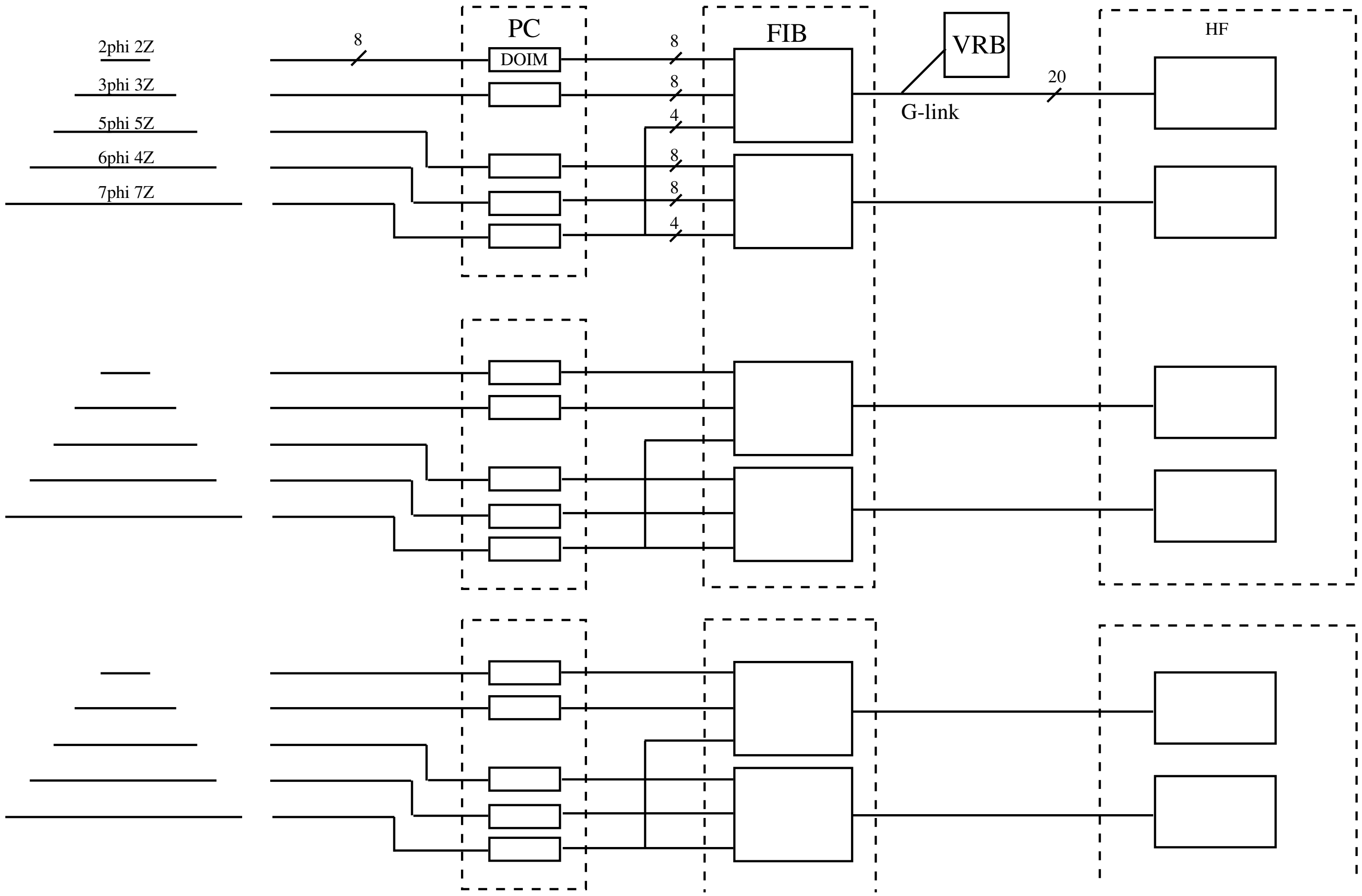
[10] SVT group, *Hit Buffer Technical Specifications*, internal note 36.

[11] SVT group, *Hit Buffer Specification and Design*, internal note 51.

[12] Yale group, *Conceptual design of the Trigger System Interface for the Run II Upgrade.*, Yale Trigger Note 1.

[13] Theresa M. Shaw, Research Division, *Specification for TRigger And Clock + Event Readout Module (TRACER).*, draft.

[14] John Conway *et al Numbering and Naming Convention for the SVX-II and ISL* CDF note 4103, March 12, 1997.



1

Figure 1: G-link organization for 1/24 of the system.