OPERATOR'S MANUAL

Model 2108

VXI Serial Data System
Digital Resource Module

Manual Revision: 1.0
Revision Date: 10/16/01
Manual Part Number: 2108RM003
Instrument Part Number:
30080/30150/30250

talon
INSTRUMENTS 150 East Arrow Hwy San Dimas, Ca. 91773 909-599-0690 Fax 909-599-6529 www.taloninst.com info@taloninst.com

CERTIFICATION

Talon Instruments certifies that this product met its published specifications at the time of shipment from the factory.

WARRANTY

Talon Instruments products are warranted against defects in materials and workmanship as follows:

(a) One year for the 2108 baseboard and all modules.
(b) Ninety days for cables and adapters.

During the warranty period, Talon Instruments will, at its option, either repair or replace products which prove to be defective.

For warranty service or repair, this product must be returned to the Talon Instruments factory. Buyer shall prepay shipping charges to the factory and Talon Instruments shall pay shipping charges to return the product to the Buyer. However, Buyer shall pay all shipping charges, duties, and taxes for products returned to Talon Instruments from another country.

Talon Instruments warrants that its software and firmware designated by Talon for use with its instruments will execute its programming instructions when properly installed on the instrument. Talon Instruments does not warrant that the operation of the instrument, or software, or firmware will be uninterrupted or error free.

LIMITATION OF WARRANTY

The foregoing warranty shall not apply to defects resulting from improper or inadequate maintenance by the Buyer, Buyer-supplied software or interfacing, unauthorized modification or misuse, operation outside of environmental specifications for the product, or improper site preparation or maintenance.

NO OTHER WARRANTY IS EXPRESSED OR IMPLIED. TALON INSTRUMENTS SPECIFICALLY DISCLAIMS THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

EXCLUSIVE REMEDIES

THE REMEDIES PROVIDED HEREIN ARE BUYER'S SOLE AND EXCLUSIVE REMEDIES. TALON INSTRUMENTS SHALL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, WHETHER BASED ON CONTRACT, TORT, OR ANY OTHER LEGAL THEORY.

SAFETY FIRST

PROTECT YOURSELF AND THE EQUIPMENT

Follow these precautions:

- Don’t bypass the chassis’ power cord’s ground lead with two-wire extension cords or plug adapters.
- Don't disconnect the green and yellow safety-earth-ground wire that connects the ground lug of the chassis power receptacle to the chassis ground terminal.
- Don't energize the chassis until directed to by the installation instructions.
- Don't repair the instrument unless you are a qualified electronics technician and have instructions from Talon Instruments.
- Pay attention to the **WARNING** statements. They point out situations that can cause injury or death.
- Pay attention to the **CAUTION** statements. They point out situations that can cause equipment damage.
- Use ESD static control procedures when handling the 2108 or any of its modules.

Table of Contents

1.0 INTRODUCTION

The Model 2108 is a flexible and capable serial bus emulator module that can be programmed to emulate a vast variety of serial bus protocols at speeds from 2kBps to 200MHz data rate. The flexibility of the 2108 module allows the user to fine-tune virtually all aspects of a serial bus interface, both at the electrical and logical levels.

The Model 2108 is comprised of a "C" size VXI motherboard which houses four instrument modules and four front-end signal conditioning modules. For serial emulation the instrument modules are the transmitter, 2108TX and the receiver, 2108RX . A number of front-end modules for the 2108TX and 2108RX are available to provide signal conditioning for the various requirements imposed by different interfaces.

The Model 2108 may be configured with 1-4, 2108TX's, 1-4, 2108RX's or a combination of the two. If a 2108TX and 2108RX are installed in channel slots 1&2 or 3&4 they may be operated in a bi-directional mode.

1.1 BASIC ELEMENTS OF SERIAL INTERFACES

There are a variety of serial interfaces being used today in the consumer, industrial and military sectors. Although a significant percentage of these interfaces are comprised of a number of standardized serial protocols, such as USB, IEEE 1394 and MIL-STD-1553, there is still an equally large percentage composed of custom serial interfaces. This group includes not only purpose-built interfaces but also variants of the standardized protocols.

In all instances, these standard and custom serial interfaces share some very basic elements. These elements include a set of electrical characteristics, a logical protocol and a method of clocking the data. In addition, some serial buses require special waveforms and control signals. The Model 2108 provides the resources in a programmable format to meet the physical and logical characteristics of most interfaces.

1.1.1 A SET OF ELECTRICAL CHARACTERISTICS

The data bit is the very building block of any serial transmission and assumes physical properties that must be defined to suit a particular protocol. The Model 2108 provides the following programmable resources to define the serial bus electrical characteristics:

a) **Signal Type**- differential, bi-polar or trinary
b) **Voltage Levels** - various levels from ECL, LVDS, TTL, +/-15Vdc, etc.
c) **Slew Rate** – min. 0.15V/ns to a max. 3V/ns
d) **Termination** – may be programmed on or off

The Model 2108 transmitter enables the user to set not only the default electrical characteristics of the bit, but also a second set which may be used to create "error" states.

1.1.2 A LOGICAL PROTOCOL

After the data bit is defined as an electrical entity, the logical protocol decides how that entity will represent logical data in the context of a serial transmission. Bit formats such as NRZ-L represent a logical "1" with a corresponding voltage "high" level and a logical "0" with a voltage "low" level. Variations of the Bi-Phase format equate either the rising or falling edges of the data pulses with a logical "1" or a logical "0", depending upon the particular variant. The Model 2108 supports all of the popular bit formats, such as:

a) **Non-Return** – Level, Mark or Space
b) **Return** – One, Compliment or Zero
c) **Bi-Phase** – Level, Mark or Space
d) **DBi-Phase** – Mark or Space
e) **AMI** - Alternate Mark Inversion

In addition to the bit format, bit ordering during a transmission is also determined by the protocol. This order can be either LSB (Least Significant Bit) or MSB (Most Significant Bit) first.

Parity generation is calculated "on-the-fly" and the Model 2108TX allows the user to insert the parity bit on a word or frame basis.

1.1.3 INTRODUCTION

The Model 2108 provides internal clock sources for each channel which supports data rates from 5kbps to 200Mbps. Additionally, the user may also select to clock the channels from external sources at the 5kbps to 200Mbps rates. In addition the 2108RX receiver may be phase locked to the external clock or to the received data.

1.1.4 WAVEFORM DATA

Some serial buses require data which cannot be generated using logical ones and zeros to be output as part of a data stream. This data is usually referred to as "invalid bit format". The Model 2108TX transmitter provides register space to define up to 4 waveforms from 2 to 12 clock periods in length that may be inserted in serial data streams. There is an additional register memory for defining up to 4 waveforms of 2 clock periods which are used to generate word or frame gaps. The 2108RX receiver's trigger logic may be programmed to recognize and trigger on waveform data as well.

1.1.5 CONTROL SIGNALS

In addition to clock and data signals some serial buses require separate signals for sync, enables and triggering. The Model 2108 provides 8 input/output signals in addition to the clock and data lines for emulating complex serial interfaces or to synchronize with other instruments in the test system . These signals are:

a) **Markers** (2)	Output from the Model 2108TX, these signals may be programmed in sync or 1-3 bit times prior or following the first and last data bits of a word or frame.
b) **Output Flags** (2)	Output from the Model 2108TX, these signals are pulses and can be used to trigger the UUT or other instruments in the test system. They are sync'd to the first bit of a word or frame and remain true until the word is output.
c) **Strobe** (1)	Output from the Model 2108TX, this signal can be used to sync to the data bits or as a clock in bi-phase format applications.
d) **Sync Pulse** (1)	Output from the Model 2108TX, this signal may be programmed to start at a bit and remain true for up to 16 bits.
e) **Input Flags** (2)	Input to the Model 2108TX, these signals may be used to start execution of a data stream or word from the Model 2108TX Transmitter.
f) **Receiver Triggers** (16)	Input to the 2108TX from the 2108RX, each trigger may be used to start execution of a unique data table.
g) **Qualifier Signals** (2)	Input to the Model 2108RX, tests for High or Low to trigger recording.
h) **Pattern Triggers** (16)	Up to 16, 32 bit wide patterns may be used to set 1 of 16 triggers to start recording of data by the 2108RX and output from the 2108TX.

2.0 SOFTWARE

The Model 2108 is shipped with two sets of software. The first is a fully functional set of VXIPlug&play drivers including a Soft Front Panel. The second is a user application package referred to as the 2108 Project Development Software.

2.1 VXIPLUG&PLAY SOFTWARE

2.2 2108 PROJECT DEVELOPMENT SOFTWARE (PDS)

The 2108 PDS is a comprehensive application package that provides the user with easy to use graphical programming tools to develop set-up files, execute tests and view the recorded data. The package contains 3 distinct application programs to provide the majority of users with all the necessary functions to develop a functioning serial interface. It was developed for the Windows environment and operates on Win95, 98, 2000, or NT based systems.

2.2.1 PROJECT DEVELOPMENT EDITOR (PDE)

The Project Development Editor is the basic application program. It provides the resources to program the logical and physical characteristics of a serial interface. In addition it provides the tools to program data tables and test sequences for stimulating the Unit Under Test, (UUT). Record triggers based on unique start patterns are also easily programmed using the 2108RX receiver's GUIs. Files created by the PDE are saved and may be downloaded using the VXI drivers or the Execution Manager.

2.2.2 EXECUTION MANAGER (EM)

The Execution Manager application provides an interactive link to download set-up files from the VXI controller to a Model 2108. In addition it allows the user to select and run any test sequences defined using the PDE. The user may loop from 1 to 32k times or run in continuous mode to aid in program debugging. 2108RX receivers may be "armed" using the EM.

2.2.3 SERIAL LOGIC ANALYZER (SLA)

The Serial Logic Analyzer application provides an interactive means of uploading recorded data from the 2108RX receiver. The data may be viewed as "raw" data or segmented by bit count, labeled and displayed as binary, hex, decimal or ASCII. This decoded data display is accomplished by defining templates that may be saved as a file. The template file may be applied to interactively recorded data when using the Execution Manager or used to analyze data recorded over a period of time. The SLA also provides search functions which operate on raw or decoded data or the label.

2.3 SOFTWARE INSTALLATION

The VXI drivers as well as the Project Development Software are contained on a CD Rom which is shipped with each Model 2108 ordered. The software may be installed on as many systems as will be used to program or control the Model 2108. The software may be distributed as part of a system shipped by Talon's customer without restriction.

2.3.1 MINIMUM SYSTEM REQUIREMENTS

- IBM-compatible PC with CD-Rom Drive
- Windows 95, 98, 2000, ME, or NT
- 64M+ RAM recommended
- Windows® compatible mouse or other pointing device

2.3.2 RUNNING THE INITIAL SETUP PROGRAM

Windows® Autoplay feature will automatically start the setup program. If Autoplay is not enabled, follow the steps below to run the setup program.

1. Click on the Start Menu and choose Run.
2. Type D:\setup.exe and press Enter. If the CD-Rom drive is not drive D, substitute the correct drive letter for the D:.
3. The menu program for the disc will start and an opening screen will appear. Click the Next button to proceed and follow the instructions on the screen.

2.4 HARDWARE INSTALLATION

After unpacking the Model 2108 visually inspect the unit for any obvious shipping damage. If any damage is noted, contact Talon Instruments immediately for advice on how to proceed.

2.4.1 BASEBOARD SWITCHES

There are two switches the user needs to check and possibly set for his system prior to installing the Model 2108 into the VXI chassis. These switches and their setting are described Section 3.1 of the Model 2108 Reference Manual.

2.4.2 INSTALLING THE 2108 INTO THE VXI CHASSIS

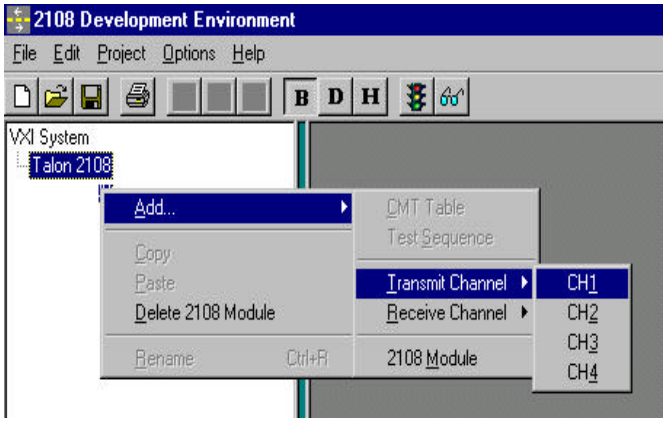
The Model 2108 may be installed in any slot other than the Slot 0 position. It should be inserted firmly but without undue force. When properly seated the front panel will be flush against the chassis card cage and the tightening screws may be used to hold the module firmly.

3.0 GETTING STARTED USING THE PROJECT DEVELOPMENT EDITOR

The Soft Front Panel may be used to determine if the Model 2108 is up and operational. Click on the 2108sfp.exe icon or run the file to start. The software will search for installed 2108 modules and report with the logical addresses.

3.1 DEFINING THE 2108 CONFIGURATION

The first step in using the PDE is to configure the 2108 (s) being programmed by defining the number and type of instruments installed in each channel slot. After starting the program, right click on VXI System in the menu directory to add a model 2108 module. Right click on the Talon 2108 module being programmed to add either Transmit or Receive channels or additional 2108 modules. See fig. below.



3.2 ROUTING THE 2108TX SIGNAL PINOUTS

The 2108TX has multiple signals which may or may not be needed for the serial interface being programmed. The I/O modules have access to all the 2108TX signals but only a limited number may be output due to the limited number of programmable drivers on the standard I/O modules (TX01 and TX02). Custom I/O modules could be designed with enough drivers to output all signals and would not require the use of the Signal Pinout directory menu.

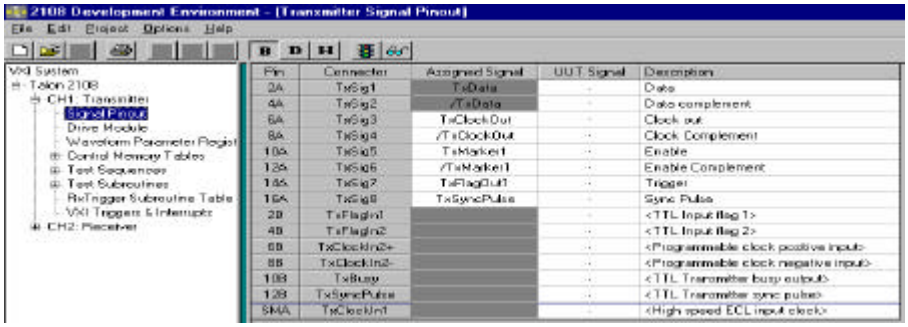
3.2.1 DATA AND CLOCK

The TxData and TxClockOut signals are always output on pins 2A and 4A if bi-polar mode is selected. If differential is selected then TxData will be on pin 2A and /TxData will be on pin 4A. Grounds will always be on the odd pins be-

tween signals.

3.2.2 AUXILIARY SIGNALS

The user selects the signals he needs for his interface and routes them to the selected pins using the pulldown selections in the Assigned Signals column. The user may also elect to enter names in the UUT and Description columns for documentation purposes.



3.3 TRANSMITTING DATA

The 2108TX module provides the necessary resources to output data in a serial stream using different formats, protocols, data rates, etc. It also provides auxiliary signals such as a shared clock, markers, sync, triggers, etc., required by many serial interfaces.

3.3.1 THEORY OF OPERATION

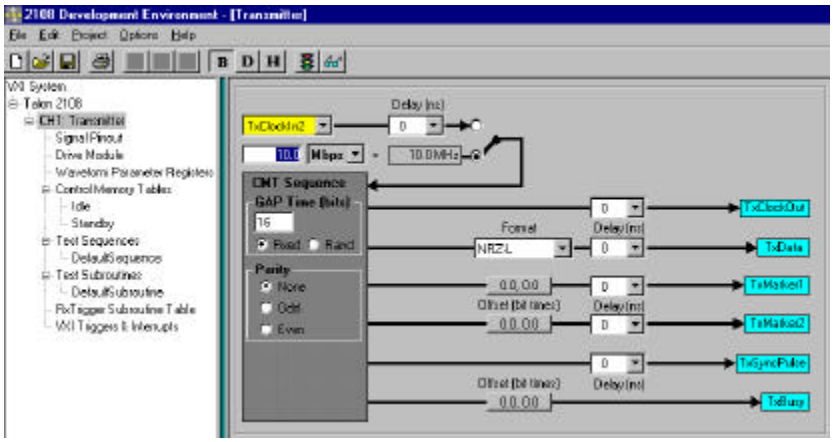
The operation of the 2108TX requires the user to define the characteristics of the serial interface to be emulated. This includes data to be output and the sequence in which the user wants the data output. It is best if the user follows a logical sequence in setting up a transmit routine. We recommend the user follow the steps listed below:

3.3.1.1 STEP 1. DEFINE THE CHARACTERISTICS OF THE OUTPUT BIT

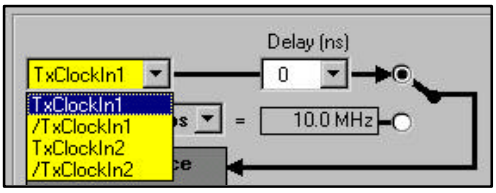
Serial data is transmitted as a stream of bits. These bits may be expressed in many formats, Non-Return, Return, Bi-Phase, etc. The bit characteristics of the serial interface are programmed using three panels of the PDE. The three panels are, Transmitter, Drive Module and Waveform Parameter Registers.

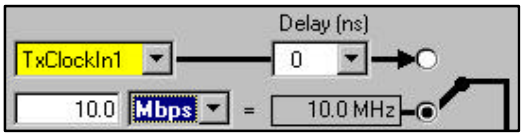
3.3.1.1.1 TRANSMITTER PANEL

The logical Bit characteristics are programmed using the Transmitter Panel.
In the Transmitter Panel the user programs:

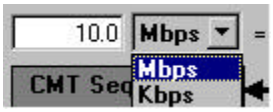


1) Clock Source- external or internal clock sources may be selected as the transmitter master clock

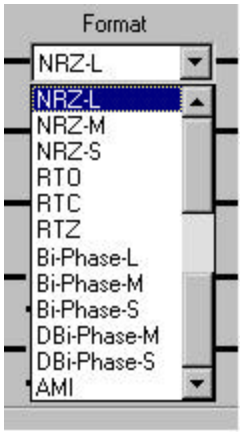




2) **Bit Rate**- bit rates from 2kBps to 200MBps may be entered.

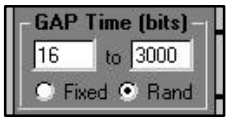


3) **Bit Formats**- the required bit format may be selected from a pull down selector

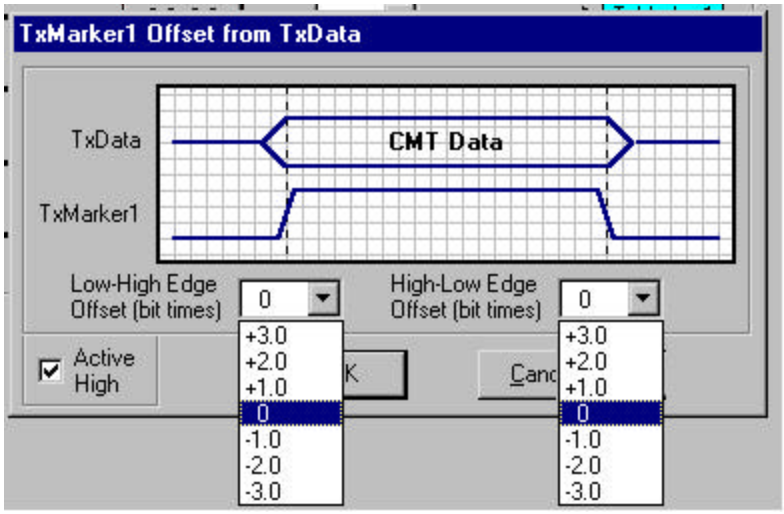


4) **Parity**- parity if required is set as None, Odd or Even

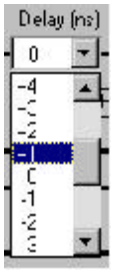
5) Gap- gap times may be set to fixed or random with limits from 4 to 65k bit times



6) Marker Signal(s)- may be set to operate coincident with the data or +/- 3 clocks from the active data and active high or low

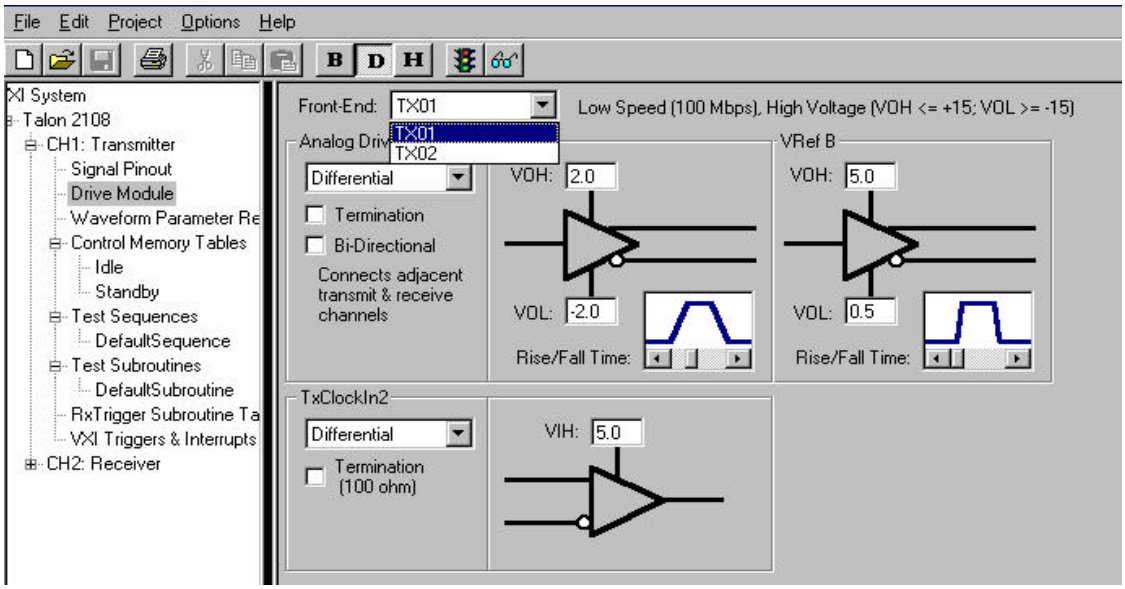


7) **Delay**- delays in 1ns increments from +/- 10ns may be programmed to most signals when cable or UUT delays cause synchronization problems



3.3.1.1.2 DRIVE MODULE PANEL

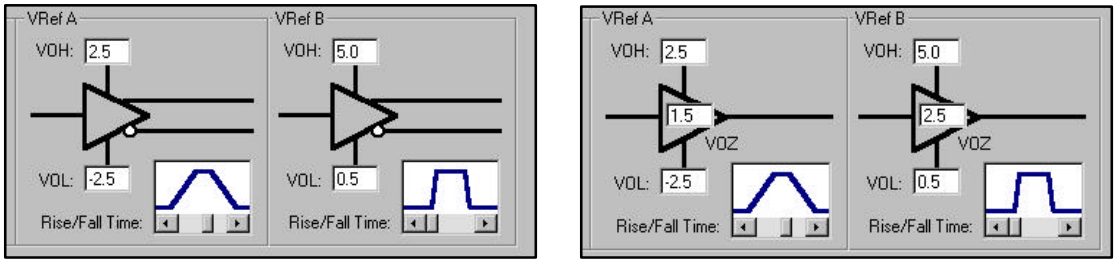
The physical bit characteristics refer to the voltage and signal type along with termination. The signals are a function of the I/O module and are programmed through the Drive Module panel. Review the manual for the specific I/O module for complete signal definitions and specifications.



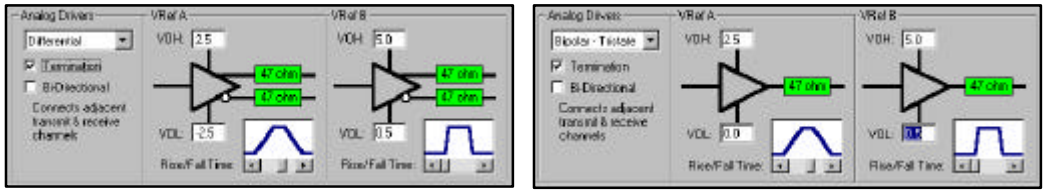
The signal characteristics to be programmed are:

1) Signal Type- the selections for most I/O modules include Differential, bi-polar with tri-state or bi-polar with a third state (trinary)

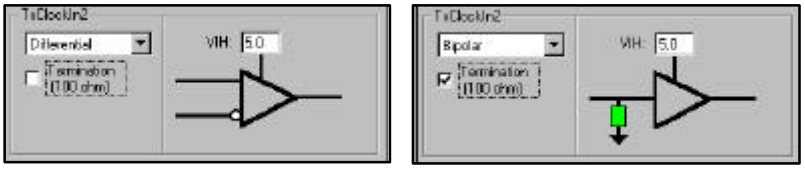
2) Signal Voltage – VrefA is the primary voltage reference requiring the output levels VOH and VOL to be entered, (3rd voltage if Bi-polar VOZ is selected), slew rate is selected by use of the slide bar window (Rise/Fall Time); VRefB may be used as a secondary voltage reference to generate errors or as a margining voltage reference



3) Termination- most of the I/O modules allow the selection of terminated or un-terminated signals.



4) TxClockIn2- if using external clock 2 as Clock Source the "signal good" level is programmed in this window for differential or bi-polar along with the termination choice

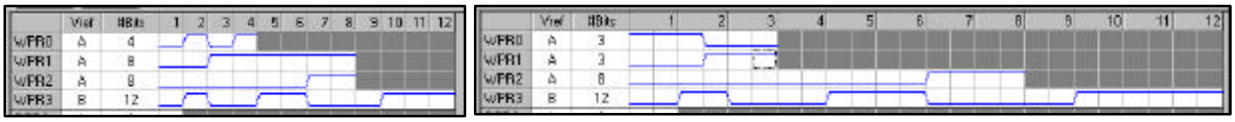


5) Bi-directional- if selected the transmitter I/O module will be the only connection to the interface bus and incoming data will be routed to the adjacent 2108RX module for bi-directional operations

3.3.1.1.3 WAVEFORM PARAMETER REGISTERS PANEL

The third aspect to bit definition is the requirement to use waveforms for data representation. Some bi-phase buses use non-logical bit patterns for sync codes. For example, the military 1553 bus has a Command Word sync code which is a signal represented as 1½ bit times high followed by 1½ bit times low. Another bus needs a pulse stream where 2 bits low out of 8 is a zero and 6 bits low out of 8 is a one. These waveform are programmed via the Waveform Parameters Register panel. There are two types of waveform registers:

1) Waveforms- 4 custom waveforms (WPR0, WPR1, WPR3 & WPR4) may be defined as using Voltage Reference A or B, the number of bit times (4 to 12 bits), and the waveform. Waveforms are programmed by selecting a cell and clicking or entering an L or H to set desired state, (bi-phase selections allow ½ bit time selections between 2 & 12 bits).



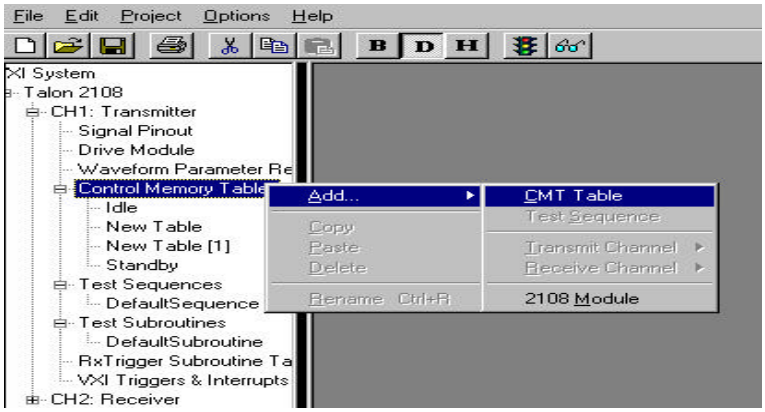
2) Gap Waveforms- there are 4 waveforms that may be used as Gap waveforms, (GRP0, GPR1, GPR2 & GPR3). They are represented as a single bit time and are programmed by selecting the Voltage Ref. A or B and the desired state, High, Low or Tri-state. Bi-phase selections allow for ½ bit time selections.

3.3.1.2 STEP 2. DEFINE THE BIT SEQUENCE(S)

All serial interfaces pack the bits in some logical sequence such that the receiver can reverse the sequence and extract meaningful data from the data stream. The schemes to logically pack data into "frames" or "packets" is unique for every interface. The 2108TX allows the programming of almost any conceivable combination of bit sequences, thereby allowing the emulation of interfaces with unique start/stop codes, parity bits, bit stuffing, as well as the generation of continuous data.

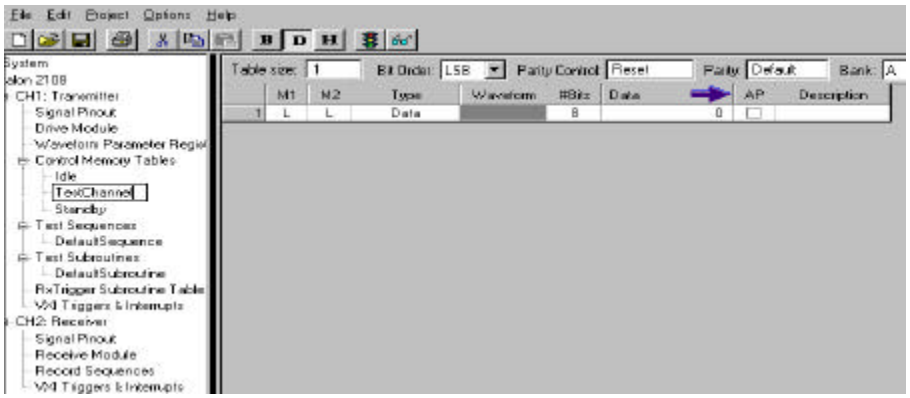
3.3.1.2.1 CONTROL MEMORY TABLES

The programming of bit sequences is accomplished in the Control Memory Table panel. This panel represents the 8Mbits of memory reserved for data to be transmitted by the 2108TX. Frames are defined as tables. A table may be just a data word or comprised of multiple data word(s), waveform(s), gaps or random data words. New tables may be added by right clicking on Control Memory Tables in the menu directory, selecting Add CMT Table and entering a name for the table.



3.3.1.2.2 PROGRAMMING CONTROL MEMORY TABLES

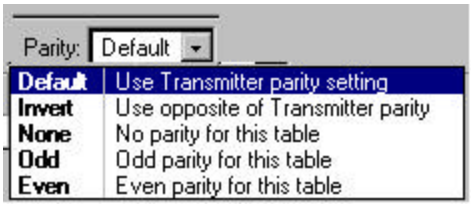
A Control Memory Table is programmed by double clicking on the table name and entering or selecting the elements which define the table characteristics. The following entries are available:



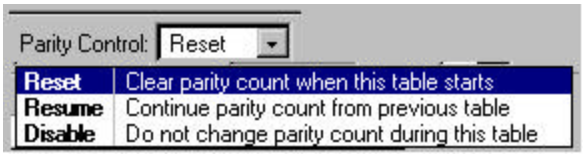
1) Table Size: enter the number of steps or words needed for the data any number up to the available memory depth may be entered.

2) Bit Order: select from the pull down window to select the order in which the data to be transmitted, Most Significant Bit or Least Significant Bit.

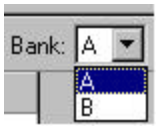
3) Parity Control: provides selections for the user to Reset the parity calculator, Resume to continue the parity calculation from a previous table, or Disable the parity calculation altogether.



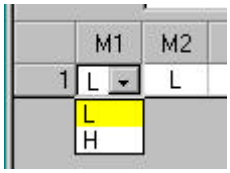
4) Parity Control: this window allows the user to select or override the default parity programmed in the Transmitter Panel.



5) Bank: the 8Mbits of memory is divided into two 4Mbit banks for ping-pong operations, the pull-down selector is used to select the bank A or B to store the table being defined.



6) M1 or M2- if the Marker signals are to be used while the step data is transmitted the level is selected as low (L) or high (H) for M1 and/or M2.



7) **Type**- the type of data to be output for each step of the table is selected in this pull-down window by clicking on one of the following choices:

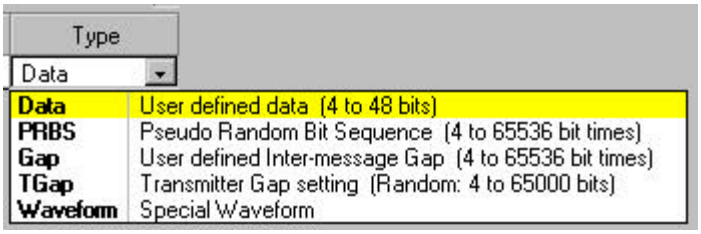
a) **Data**- fixed data from 4 bits to 48 bits in length

b) PRBS- pseudo random bit sequence from 4 to 65k bits

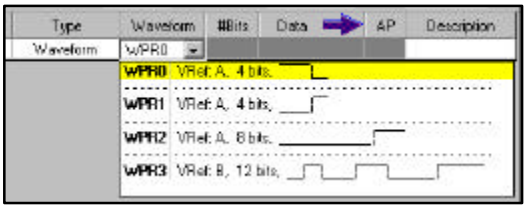
c) **Gap**- fixed intermessage gap using one of the Gap Waveforms from 4 to 65k bit times

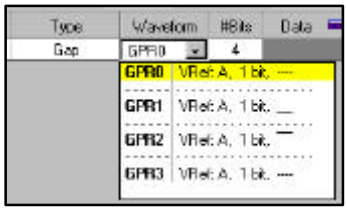
d) **TGap**- uses the default Transmitter Panel setting (random or fixed) and one of the Gap waveforms

e) Waveform- one of the 4 waveforms defined in the Waveform Parameters Register panel will be selected.



8) Waveform- if Waveform, Gap or TGap was selected as the Type of data to be transmitted in a step this window will be active and allow the selection of one o f the previously defined waveforms.



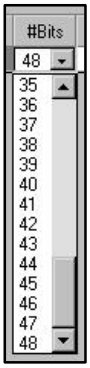


9) #Bits: allows the entry of the number of bits represented in a step:

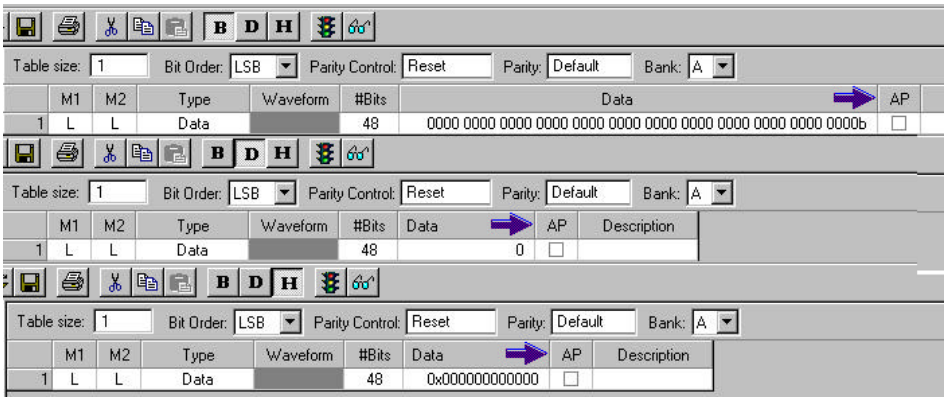
a) **Data**- the number of bits is 4 to 48 (words longer than 48 bits are entered in consecutive steps and will be output as a single stream)

b) **PRBS**- the number of bits is from 4 to 65,536 of random data

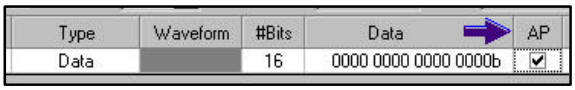
c) **Gap**- the number of Gap periods is from 4 to 65,536 bit times

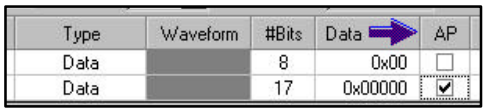


10) Data: fixed data is entered in this window as binary, decimal or hex.



11) AP: clicking on this selection box will append a parity bit to the data using the parity control selected for this table, (parity may be calculated across any number of words by selecting AP only on the last data step in the table).



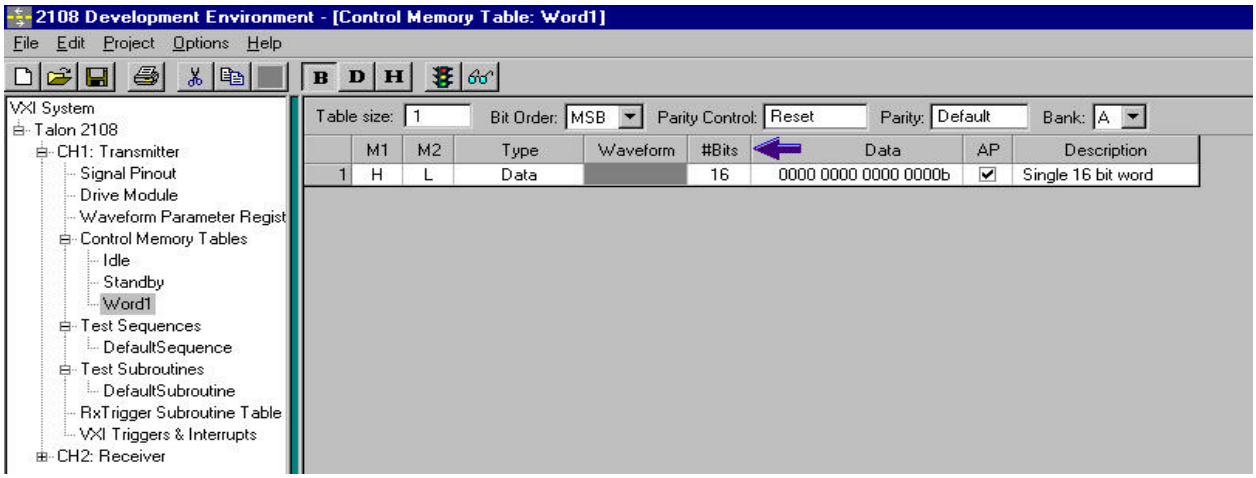


3.3.1.3 EXAMPLES OF CMT'S

3.3.1.3.1 EXAMPLE 1. SIMPLE 16 BIT ONE WORD TABLE

In example 1 the user needs to output a single 16 bit, one word table, using M1 high as a write enable signal, MSB as the bit transmission order and adding a parity bit to the data. This type of data table is easily programmed by following these steps:

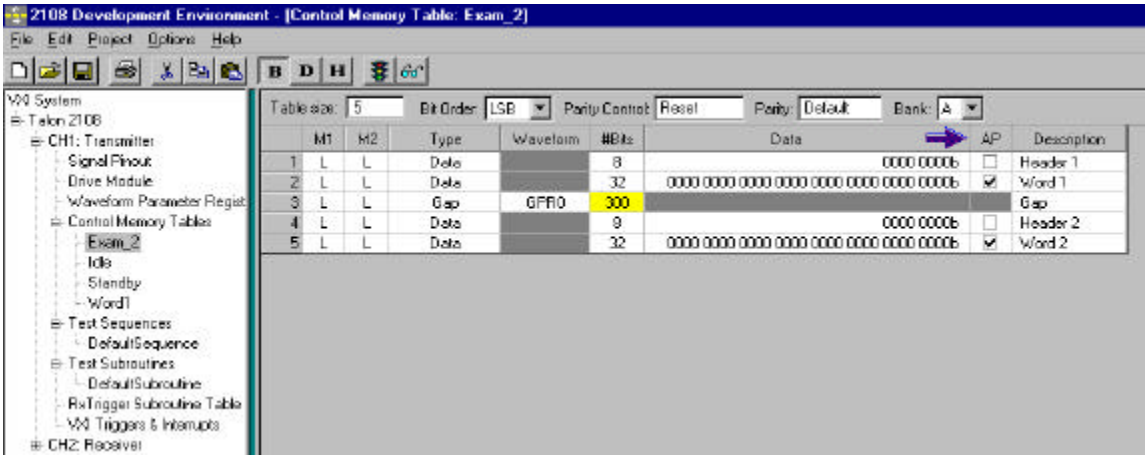
a) Right click on Control Memory Tables in the menu directory
b) Enter a name for the table, (Word1 for this example)
c) Leave 1 as the step size
d) Select MSB for the Bit Order
e) Select H for M1
f) Select Data as the Type
g) Enter 16 as the #Bits
h) Enter the data (0101111101011010b in this example)
i) Click on the AP box to add a parity bit
j) Add a description in the Description box if desired



3.3.1.3.2 EXAMPLE 2. TWO 40BIT WORDS WITH GAP

In example 2 the user wishes to output two 40bit words, (8 bit header followed by 32 bit data) separated by a 300 bit time fixed gap. Data is to be transmitted as LSB and parity is to be appended to the data. The steps are:

a) Right click on Control Memory Tables in the menu directory
b) Enter a name for the table, (Exam_2 for this example)
c) Enter 5 as the step size
d) Select LSB for the Bit Order
e) For Step 1 select Data as the Type
f) Enter 8 as the #Bits
g) Enter the data for header 1 (00001111b in this example)
h) For Step 2 select Data as the Type
i) Enter 32 as the #Bits
j) Enter the data for word 1 (01110011110010100111001111001010b in this example)
k) Click on the AP box to append the parity bit
l) For Step 3 select Gap as the Type
m) Select GPR0 as the waveform to be transmitted as the Gap signal
n) For Step 1 select Data as the Type
o) Enter 8 as the #Bits
p) Enter the data for header 1 (00001111b in this example)
q) For Step 2 select Data as the Type
r) Enter 32 as the #Bits
s) Enter the data for word 1 (01110011110010100111001111001010b in this example)
t) Click on the AP box to append the parity bit

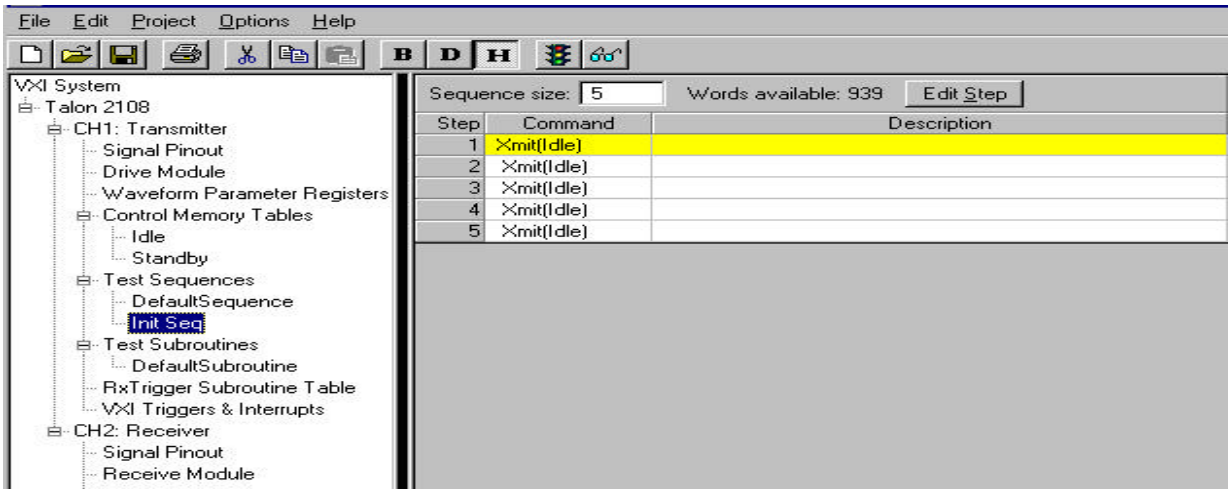


STEP 3. DEFINE THE FRAME OUTPUT SEQUENCE(S)

Once the bit format and data frames are defined, the 2108 allows the user to output these frames in any sequence he chooses. This is accomplished by programming the Bit Slice Sequence Controller using the panel labeled Test Sequences. Data may be transmitted in a single cycle, looped, or run continuously. Test sequences may be programmed to run in an automated mode, testing input signals or the contents of input data to start or stop sequences.

3.3.1.4 DEFINE THE TEST SEQUENCES

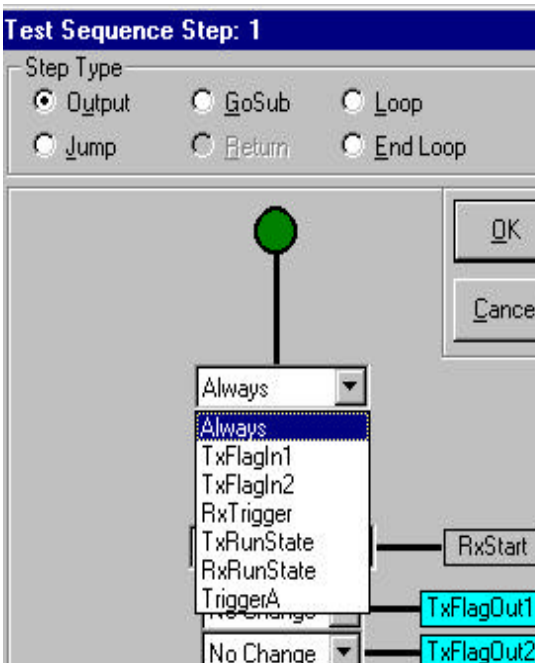
Step 3.
Test sequences are named by right clicking on Test Sequences in the menu directory. Test sequences can be referenced by any name the user wishes. Multiple test sequences can be defined to the limit of the test sequence memory. Start the process by naming a new test sequence and entering an estimated number of steps needed to perform the sequence required in the Sequence Size: window. It is usually better to guess high as the exact size may be reset after programming the sequence. The Words Available window will keep track of the memory space. After naming a Test Sequence the step list panel for the Test Sequences will appear. The Edit Step button provides access to the programming panel to perform functions from simple outputs to more complex interactive functions involving conditional and unconditional, jumping, sub-routines, looping, etc.



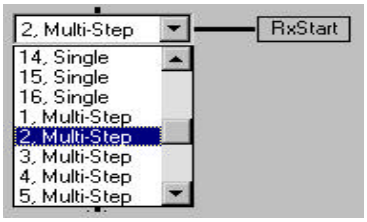
3.3.1.4.1 PROGRAMMING SEQUENCE STEPS

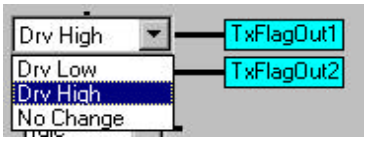
Test sequence programming is performed by highlighting the Step to be programmed and then clicking on the Edit Step button. The programming panel will appear. There are six command types that can be programmed. The six are:

3.3.1.4.1.1 OUTPUT- this command allows the user to output a single table from the Command Memory Tables list. The table may be output Always when the sequence is executed or the output triggered by an event. Events may be triggers from the UUT, a trigger from an adjacent 2108RX, run state of the 2108TX or the adjacent 2108RX, or a Trigger from the VXI bus. The test sequence may be programmed to Wait for an event to be true before proceeding or if false skip this step and proceed to the next step.

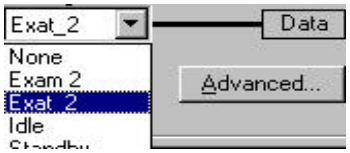


The RxStart selection window is used to arm an adjacent 2108RX to trigger on a specific trigger once or multiple times. The second and third windows set the state of the transmit flag out signals. The signals may be individually programmed to drive high or low coincident with the CMT being output.

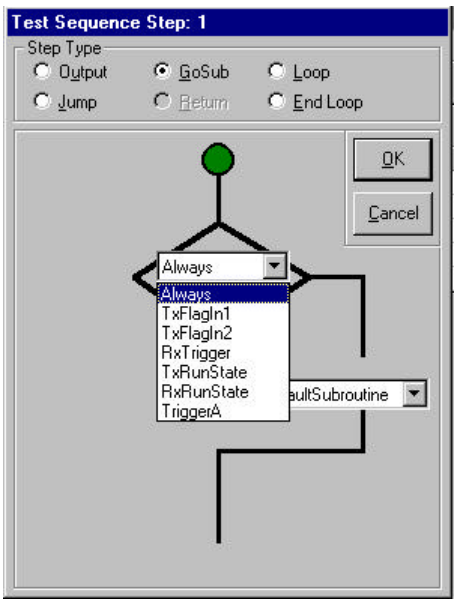




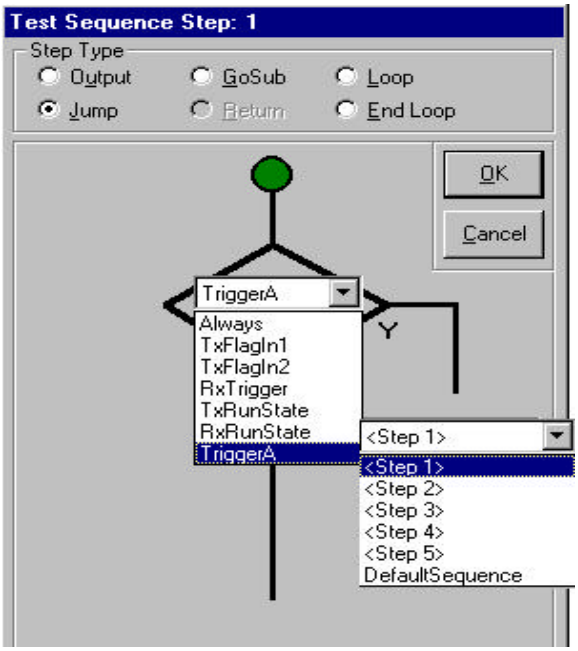
The last window in this panel is the CMT selection window for the CMT to be output.



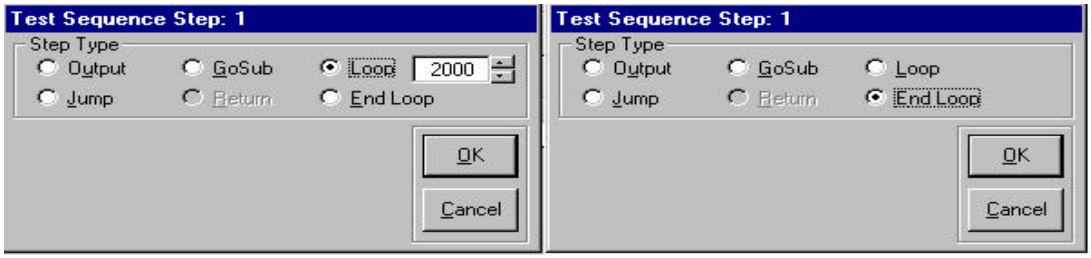
3.3.1.4.1.2 GoSub: This command allows the user to call a test subroutine. Test Subroutines have an implied Return after execution. The selected subroutine may Always be output or the trigger test functions may be programmed the same as the Output command. The selected sub routine must be defined and stored under the Test Subroutines section of the menu directory.



3.3.1.4.1.3 Jump: this command allows the user to program a Jump to a specific step in the test sequence or to a another sequence.

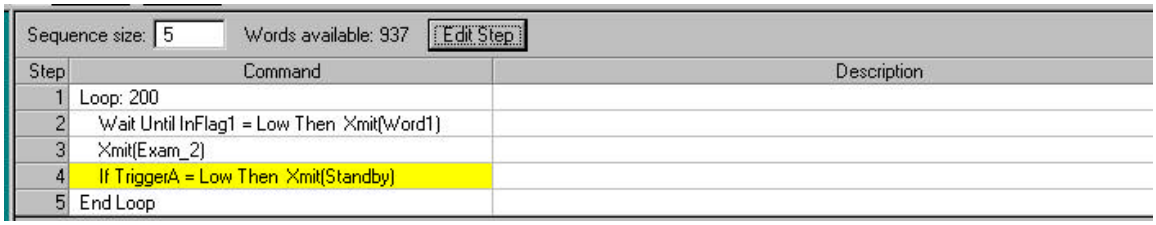


3.3.1.4.1.4 Loop/End Loop: these commands allow the user to program loops from 1 to 32k times for all sequences between the Loop command and the End Loop command. In addition, Loops may be nested 3 levels deep.



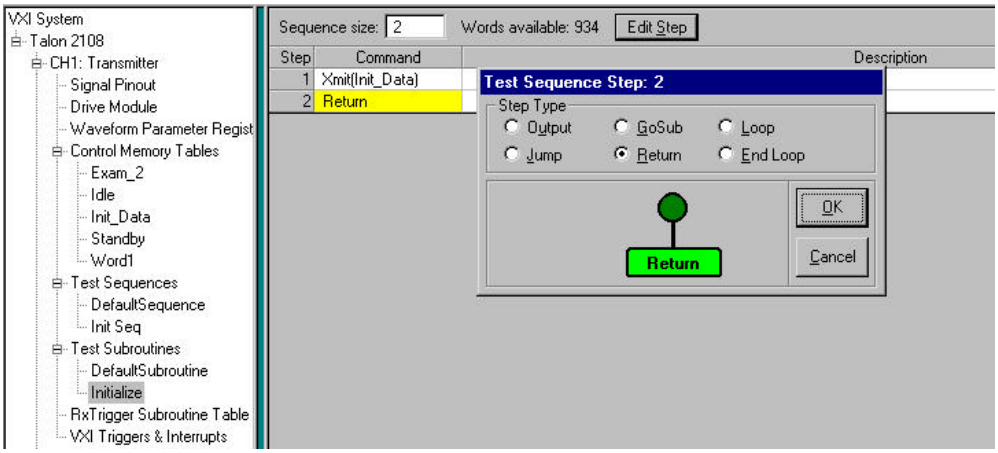
3.3.1.4.1.5 SEQUENCE STEP LIST

Once a Test Sequence is completed the Step List provides the user with a complete listing of the program for documentation in an easy to read and understand format. There is also an entry area for descriptions.



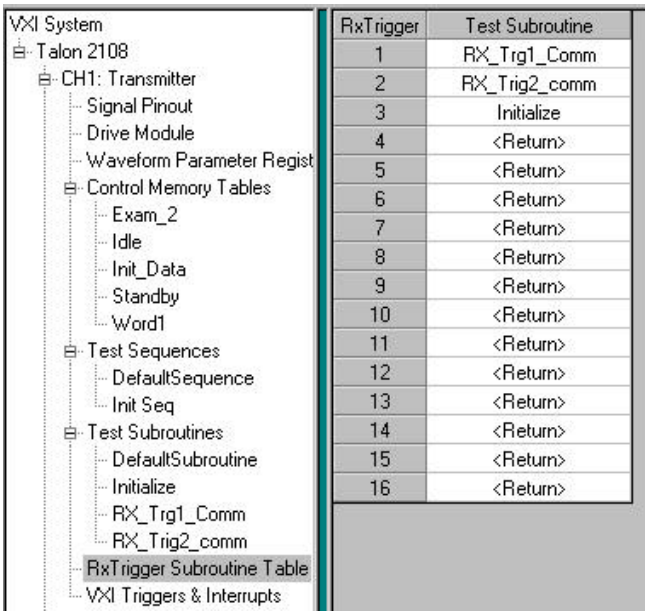
3.3.1.4.2 TEST SUBROUTINES

Test Subroutines are test sequences programmed as subroutines to be called from the main test sequences. This feature saves test sequence steps. Test Subroutines are named and programmed the same as Test Sequences except the Return command is activated and must be programmed as the last step in the subroutine.



3.3.1.4.3 RXTRIGGER SUBROUTINE TABLE

RXTrigger Subroutine Table is a subset of the Test Subroutines. A 2108TX subroutine may be triggered by one of the 16 2108RX triggers. 2108RX triggers are set when the 2108RX receives a specific UUT qualifier signal or data pattern. The unique trigger number is passed to the 2108TX and may be used to initiate an RXTrigger Subroutine Table output. This function is often used to perform a command/response operation.

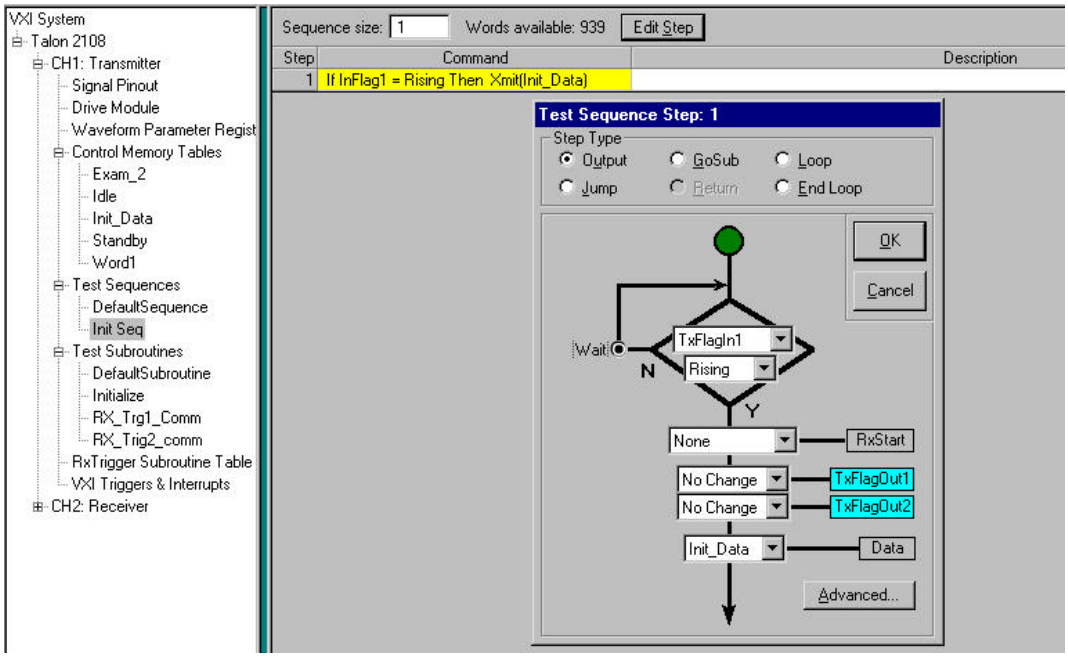


3.3.1.4 Test Sequence Examples

3.3.1.4.1 Example 1. Transmitting a CMT When Triggered

In this example the user needs to accomplish something very simple, wait for an input trigger from the UUT prior to transmitting a CMT. The example test sequence will be named, “Init Seq” and it is to output CMT, “Init_Data” when triggered by the rising edge of TXFlagIn1. Programming steps are:

a) Right click on Test Sequences in the menu directory
b) Enter a name for the table, (Init Seq for this example)
c) Leave 1 as the Sequence Size
d) Click on Edit Step
e) Select Output as the Step Type
f) Select TxFlagIn1 in the decision tree
g) Select Rising as the test
h) Click on Wait to set the 2108TX to wait for the rising edge
i) In the Data pull down window select Init_data as the CMT to be transmitted



3.3.1.5 VXI Triggers & Interrupts

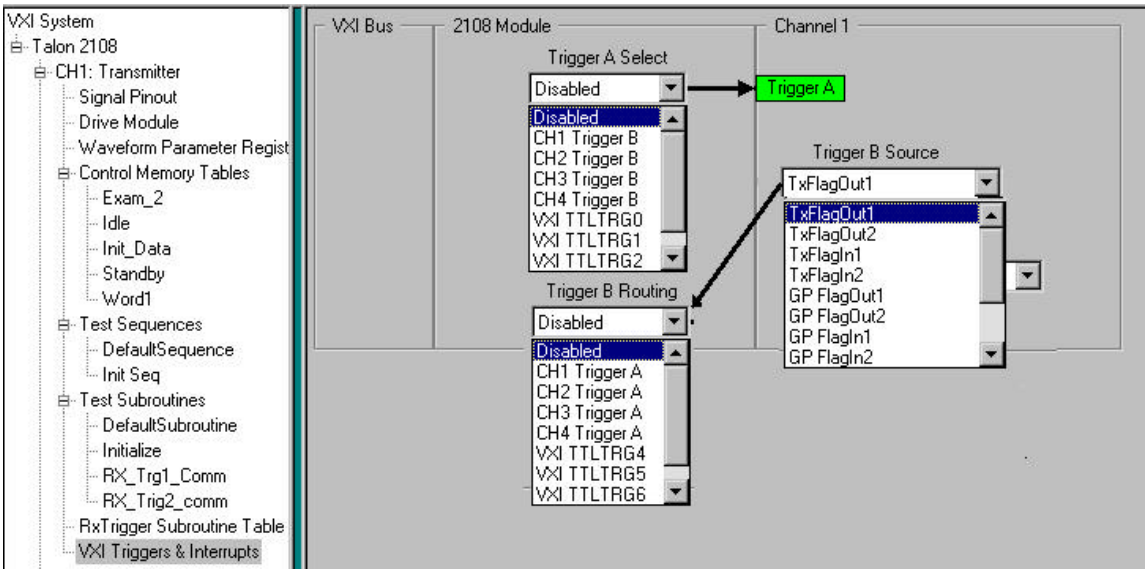
VXI Triggers & Interrupts menu directory provides for mapping of input and output triggers to/from the 2108TX.

3.3.1.5.1 VXI Triggers

The VXI triggers, TTLTRG0, TTLTRG1, TTLTRG2, or TTLTRG4 may be mapped to the 2108TX as the source for TriggerA. In addition, Trigger B from a 2108TX in one of the other 3 channel slots may be routed to TriggerA as the source. TriggerB may be sourced from one of many inputs (transmitter flags in or out, GP flags, etc.) and routed to one of the VXI triggers, TTLTRG4, TTLTRG5, TTLTRG6 or TTLTRG7or to one of the other three channel slots. In addition, TriggerB may be routed to TriggerA of Chan1, Chan2, Chan3 or Chan4.

3.3.1.5.2 VXI INTERRUPTS

There are 24 functions in the Model 2108 which could generate interrupts for the VXI controller. This window allows any one of the 24 to be set as an interrupt. The interrupt service loop must be written in the test program and enabled and disabled as needed. Select an interrupt by using the pull down list and clicking on the function which is to generate the interrupt.



4.0 CHARACTERISTICS OF A SIMPLE SYNCHRONOUS SERIAL BUS INTERFACE

Although there are many synchronous serial interface structures in use, most contain 3 common signal elements in order to ensure proper and reliable transmission of data to a destination. These elements are CLOCK, DATA, and ENABLE.

CLOCK
The CLOCK signal produces a continuous clock waveform and serves to synchronize the occurrence of each DATA bit with a specific and constant time period. Typically each bit of DATA corresponds directly to one complete CLOCK cycle.

DATA
The DATA signal transmits the actual data payload to the destination, along with other elements such as headers and error-checking information (i.e., parity).

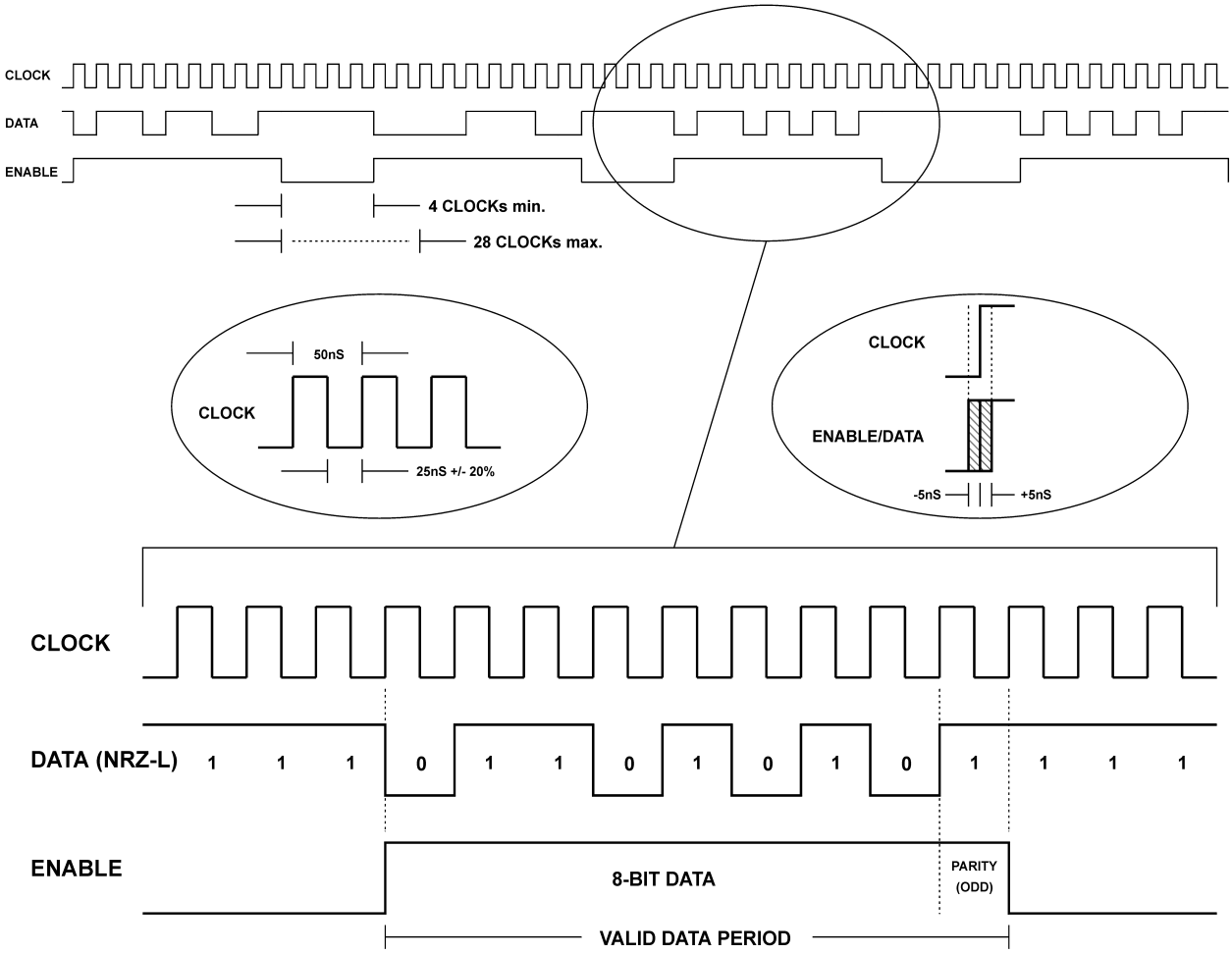
ENABLE
The ENABLE signal identifies the occurrence of a DATA transmission and can track either each specific DATA word (i.e., DATA ENABLE) or a group of DATA words (i.e., FRAME ENABLE). In all cases, DATA is considered valid only while the ENABLE signal is “true”.

4.1 AN EXAMPLE OF A SYNCHRONOUS SERIAL BUS INTERFACE

The first step in emulating any interface is to understand what elements are involved and what resources can be assigned to them. This step also includes understanding the relationships between

these elements as they perform in a real-world situation. For the particular serial bus in this example, it is known that there will need to be output channels for the CLOCK, DATA and ENABLE signals. The bus uses 422/485 type drivers. Looking at the timing diagrams below, the following relationships can be surmised:

a) The CLOCK frequency is 20MHz with a 50% duty cycle (25nS “low” period +/-20%).
b) ENABLE goes “high” to indicate the valid DATA period.
c) ENABLE and DATA transition upon the “rising edge” of CLOCK within +/-5nS.
d) Each DATA bit equates to one CLOCK period.
e) The valid DATA word length is 8 bits + 1 odd parity bit (i.e., 9 CLOCK periods total).
f) CLOCK is continuous and free-running.
g) DATA is held at a logical “high” during ENABLE “low”.
h) “Invalid” DATA periods are variable (i.e., when ENABLE is “low”) from a minimum of 4 CLOCK periods to a maximum of 28 clock periods.
i) Data Format is NRZ-L.



Example 1: Serial Interface Timing Diagram

4.2 USING THE PROJECT DEVELOPMENT SOFTWARE FOR THE EXAMPLE SERIAL BUS

Using the Model 2108TX Project Development Software, a user can easily implement a variety of serial bus emulation scenarios. Configuring setup parameters and constructing data streams are accomplished via graphical panels, whose titles are ordered in an intuitive top-to-bottom arrangement in the main project panel (shown below). We'll use this software to develop our example bus.

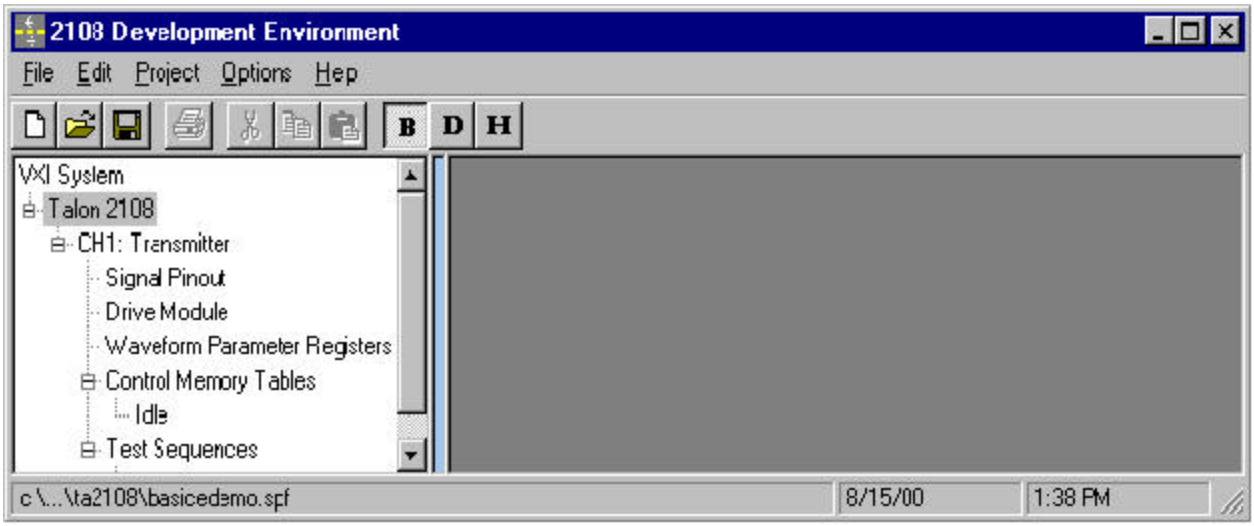


Fig. 1 2108TX Project Development Master Panel

4.2.1 PROGRAMMING THE ELECTRICAL CHARACTERISTICS

The first step in setting up the example serial interface is to program the electrical characteristics. Since this is a 422/485 serial interface we know the driver characteristics include differential signals, 0-5Vdc for the logic levels and with a low to mid-level slew rate. The Drive Module panel is used to program the electrical characteristics for both Ref. A (primary) and Ref. B (error).

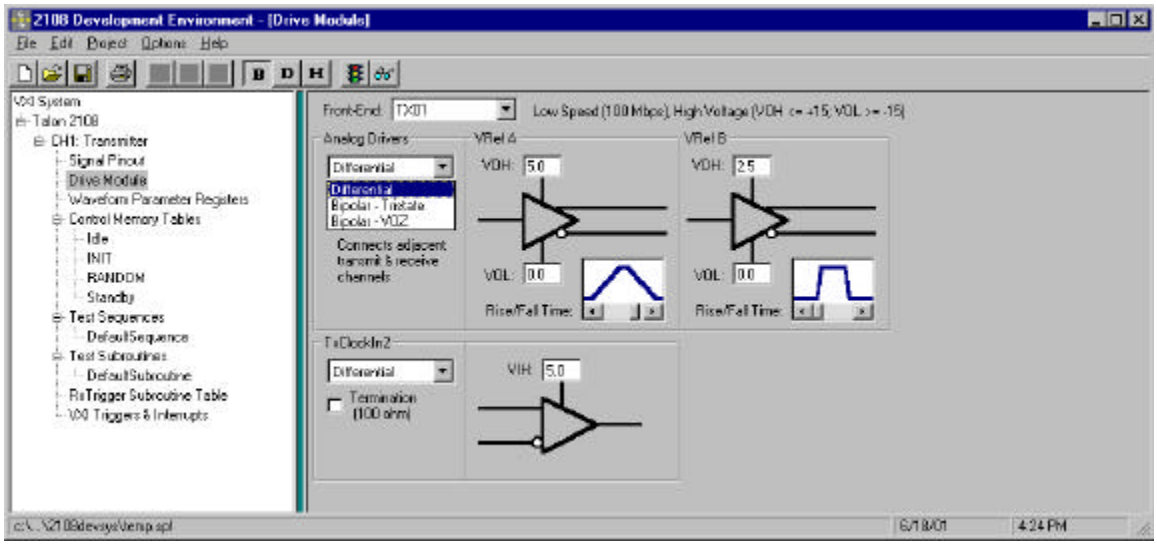


Fig. 2 Using the Drive Module Panel to Program the Electrical Characteristics

4.2.2 PROGRAMMING THE LOGICAL CHARACTERISTICS

The logical characteristics of the example are programmed by use of the Transmitter panel. Using the information from our timing diagrams and the fact that it is a 422/485 interface we know the data rate, data gap min/max values, logical format and parity type.

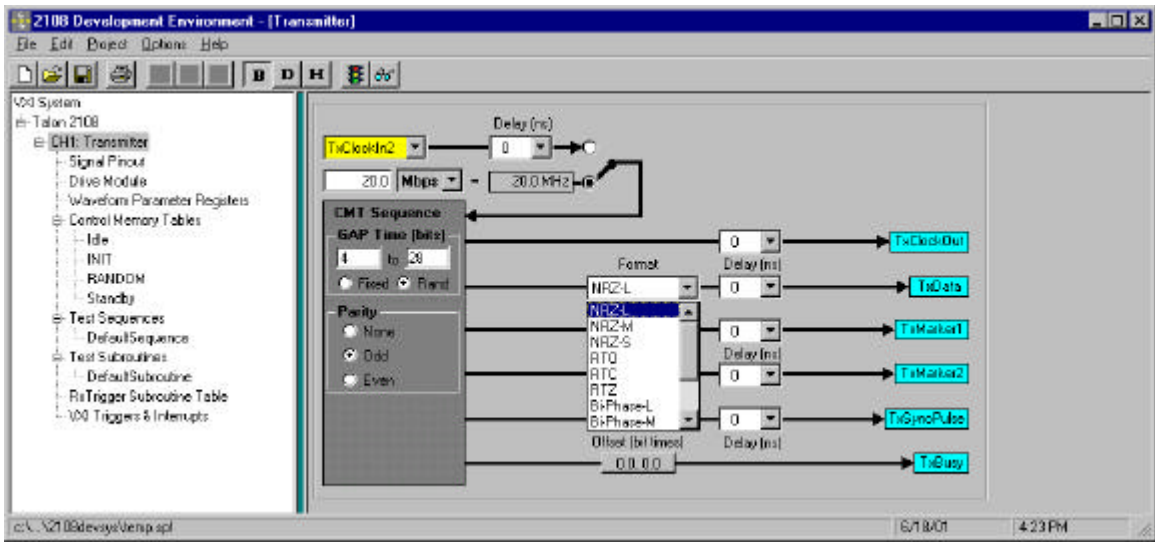


Fig. 3 Using the Transmitter Panel to Program Format, Data Rate, Gap and Parity

4.2.3 PROGRAMMING DATA

The next step after programming the electrical and logical characteristics is to prepare data files to be transmitted to the UUT. Data files are stored as Command Memory Tables, (CMT's). Each command memory table is programmed as a file containing the data and the parameters pertinent to that data file. Since our example is quite simple our first CMT will be named INIT and contain 5, 8 bit words. The bit order is set to MSB and the TGap generator is programmed to use gap waveform GPRO. Parity is appended after each word. Marker 1 (M1) is programmed to be output High during each data word transmission and Low during gap periods to match the Enable Signal operation.

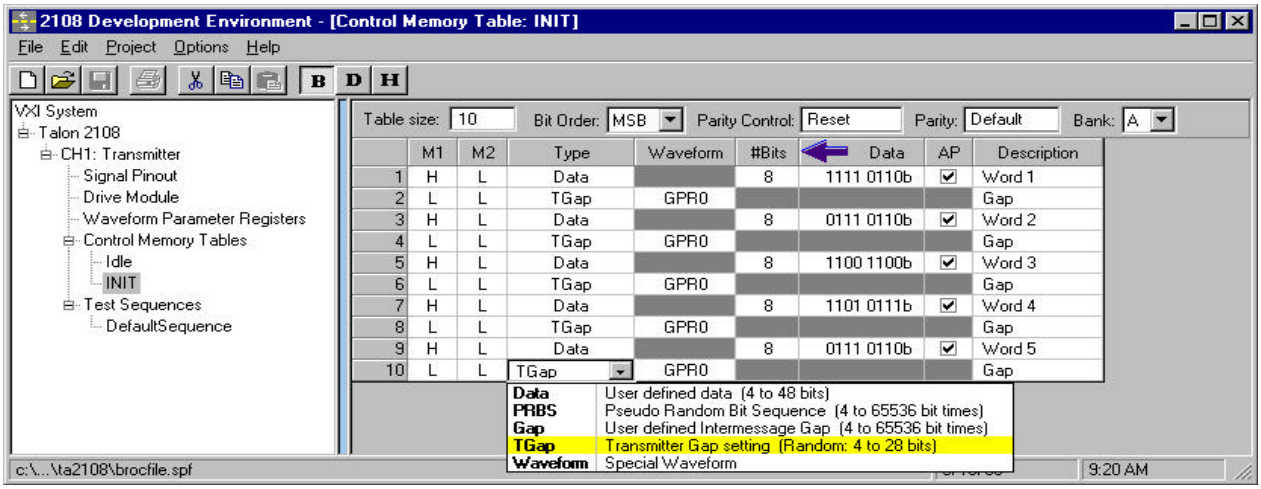


Fig. 4 Programming the Data File INIT with 5 Words Separated by Random Gaps

4.2.4 PROGRAMMING A PRBS DATA TABLE

The second data table to be used in our example will be programmed to output Pseudo Random Bit Sequence data instead of fixed data. The Control Memory Tables panel will be used to create a table named RANDOM.

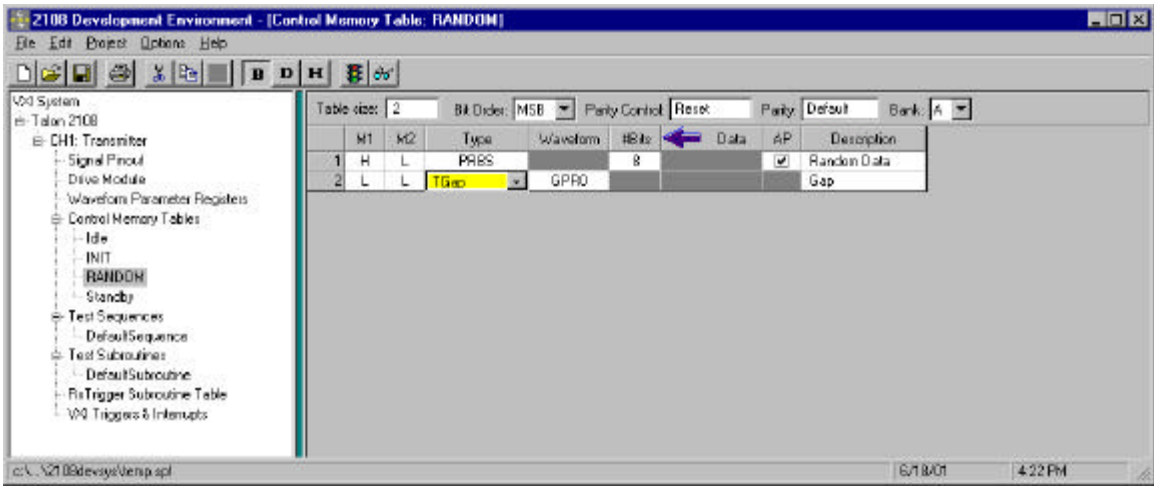


Fig. 5 Programming the Data File RANDOM with 1, 8-Bit Random Word

4.2.5 PROGRAMMING THE OUTPUT SEQUENCE

The next step in programming our example serial bus is to define the output sequence of the data files we have created. Output sequences are defined in the panel named Test Sequences. In our example we will define a simple sequence to output the CMT named INIT followed by a loop sequence outputting the CMT RANDOM looped 100 times. The Test Sequence name is START.

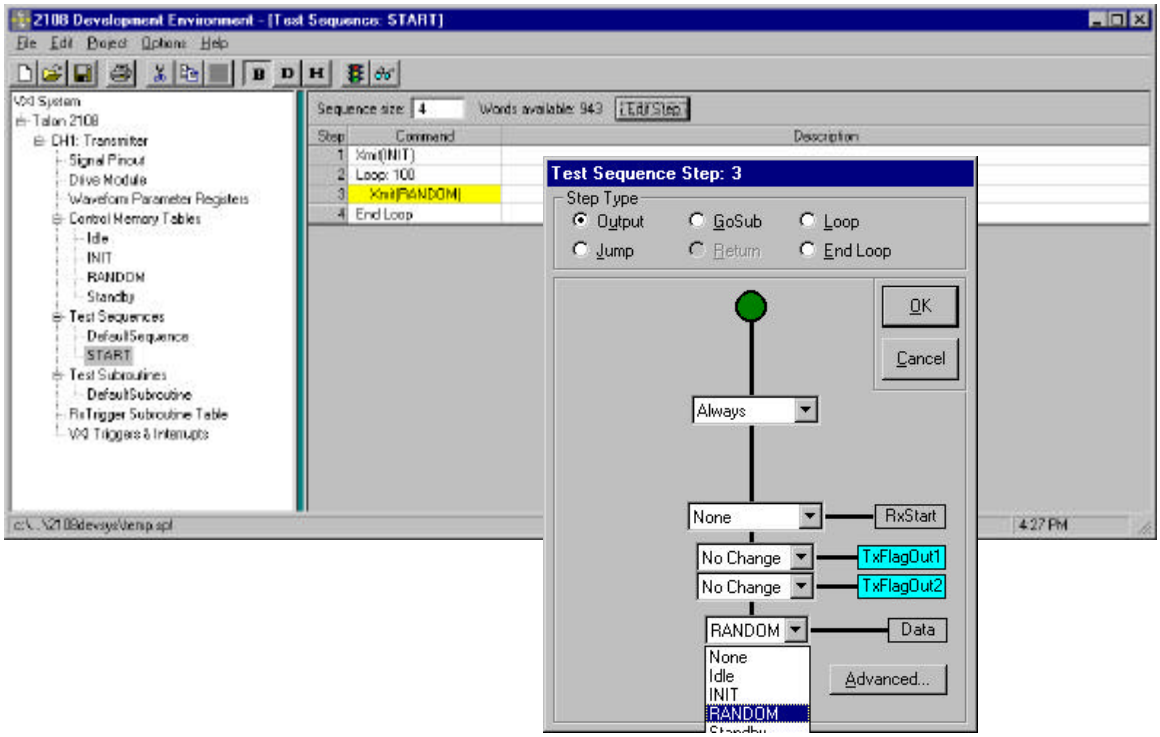
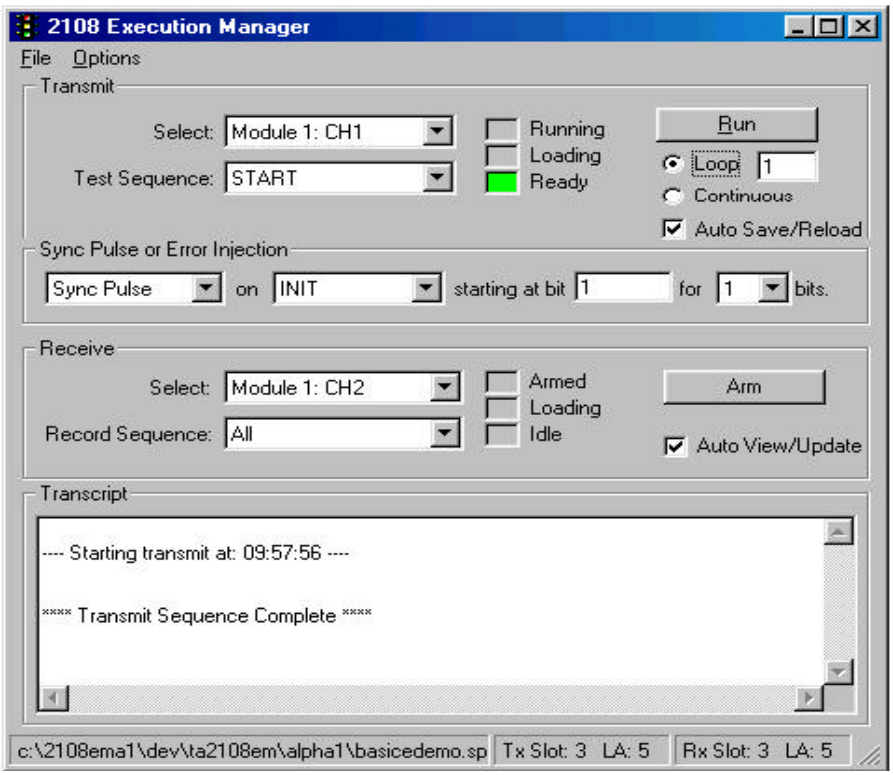


Fig. 6 The Graphical Programming Panel is Used to Program Test Sequence START

4.3 EXECUTING A CMT OR A TEST SEQUENCE

The final step in the creation of a serial bus emulation test is to connect to the "real" thing. After performing the physical interconnect the CMT's and/or the Test Sequences are downloaded to the Model 2108TX from the VXI controller. This may be accomplished in one of two ways. In the first method, the saved file created by the Project Development Editor may be interactively downloaded using the 2108TX Execution Manager. The Execution Manager is a part of the Project Development Editor. The Execution Manager provides interactive panels to download, start and run individual CMT's or Test Sequences. Control is provided for single pass or continuous loops to aid in debugging problems. The second method is use the 2108TX VXIplug&play drivers to download and control the execution of the CMT's or the Test Sequences. The Execution Manager and the VXIplug&play drivers are WIN95, 98 and NT compatible.



5.0 USING THE 2108RX RECEIVER

The 2108RX was developed to receive and record serial data at rates to 200Mbps. The architecture of the 2108RX is best described as a 200MHz logic analyzer dedicated to serial applications.

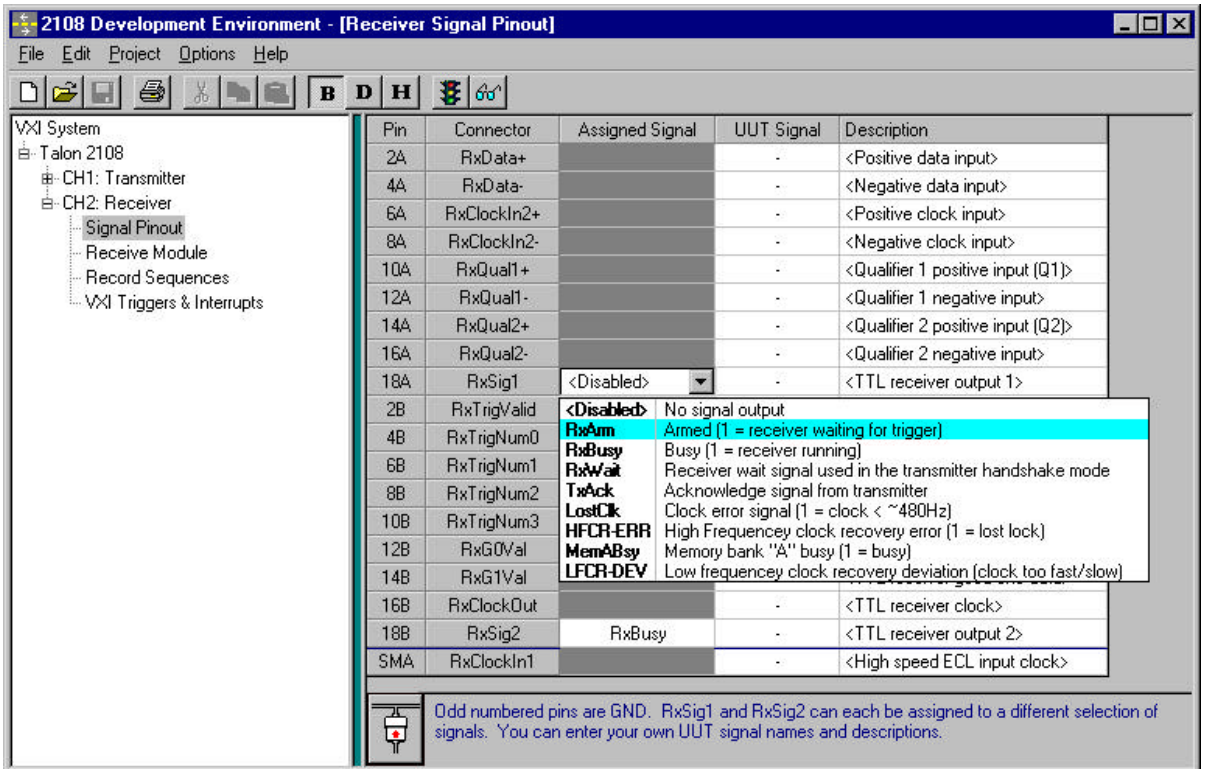
A total of 8Mb of data can be recorded (actually, there are two 8Mb memories, one storing “GOOD 1” and the other storing “GOOD 0” data). Alternately, the 2108RX can operate in a continuous recording mode where 4Mb are actively recorded while the previous 4Mb of recorded data are transferred to a mass storage media. Seamless recording is achieved by ping-ponging between the two 4Mb memory banks.

5.1 ROUTING THE 2108RX SIGNAL PINOUTS

The 2108RX provides two pins which are user assignable through the Signal Pinout panel. These pins are 18A and 18B. The signals which may be assigned to the pins are :

1. RxArm
2. RxBusy
3. RxWait
4. TxAck
5. LostClko
6. HFCR-ERR
7. MemABsy
8. LFCR-DEV

The remainder of the signals are not assignable by the user on the standard I/O modules, (RX01 and RX02). The Signal Pinout panel also allows the user to enter UUT pin names and descriptions for each pin for documentation purposes.



5.2 THEORY OF OPERATION

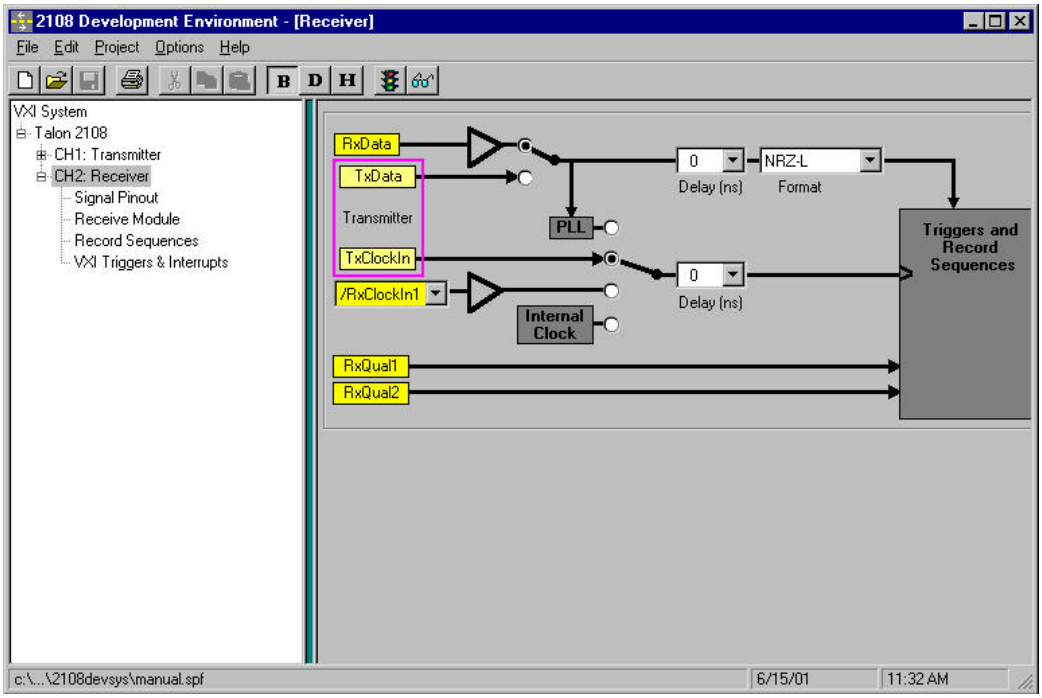
The operation of the 2108RX module requires the user to define the characteristics of the serial interface to be emulated. This includes the initiation of data recording as well as the bit characteristics, clock speed and other features of the serial interface to be emulated.

5.2.1 STEP 1. DEFINE THE CHARACTERISTICS OF THE SERIAL BIT

The characteristics of the serial bit include the logical and physical characteristics as well as the clocking of the data.

5.2.1.1 RECEIVER PANEL

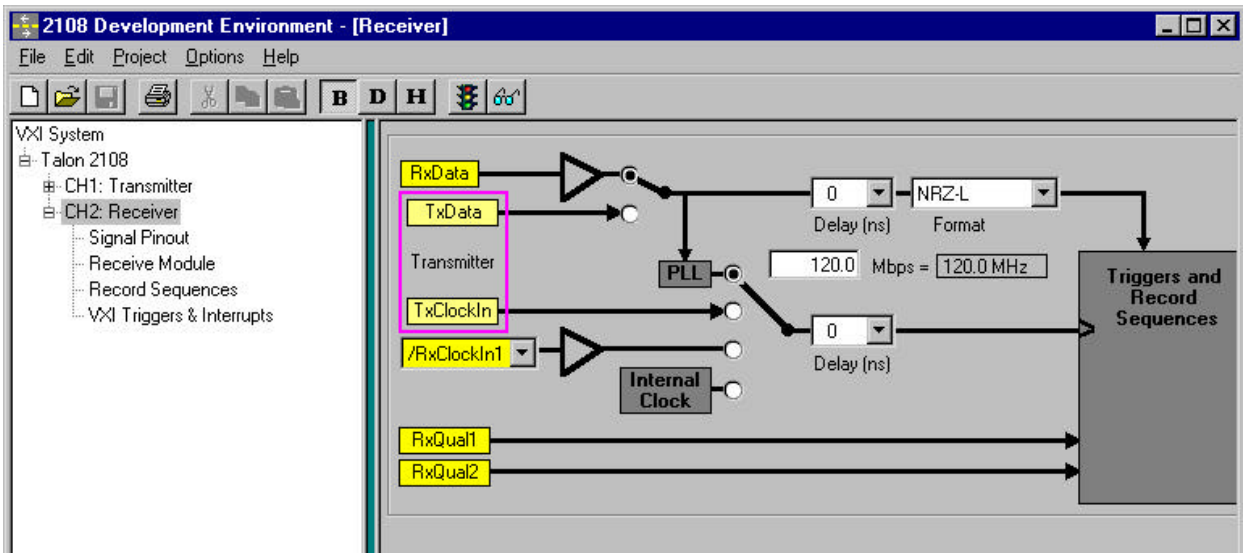
The logical bit characteristics and clocking information are programmed in the Receiver Panel.



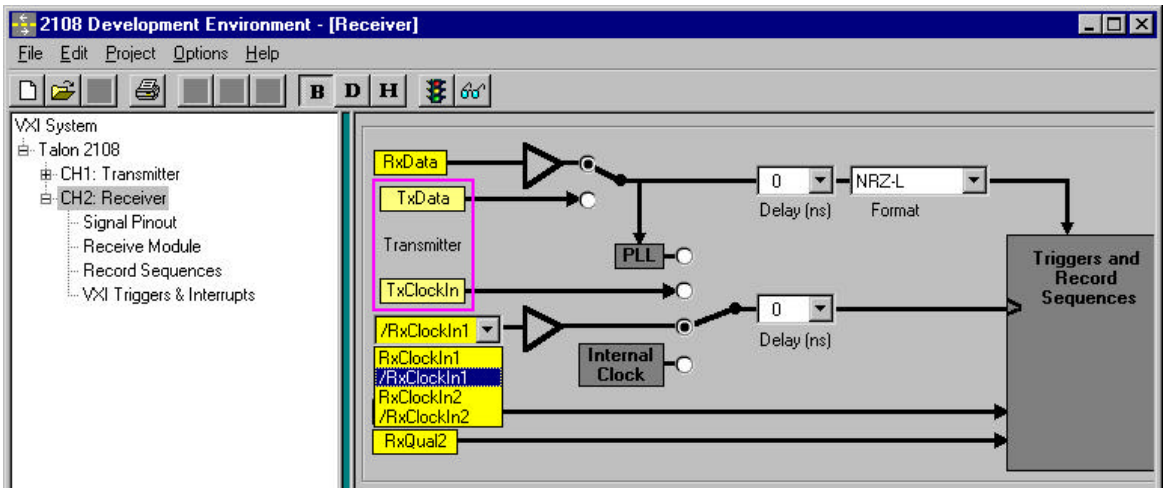
In the Receiver panel the user programs:

1) Clock Source- external or internal clock sources or the data may be selected as the source for the Receiver clock.

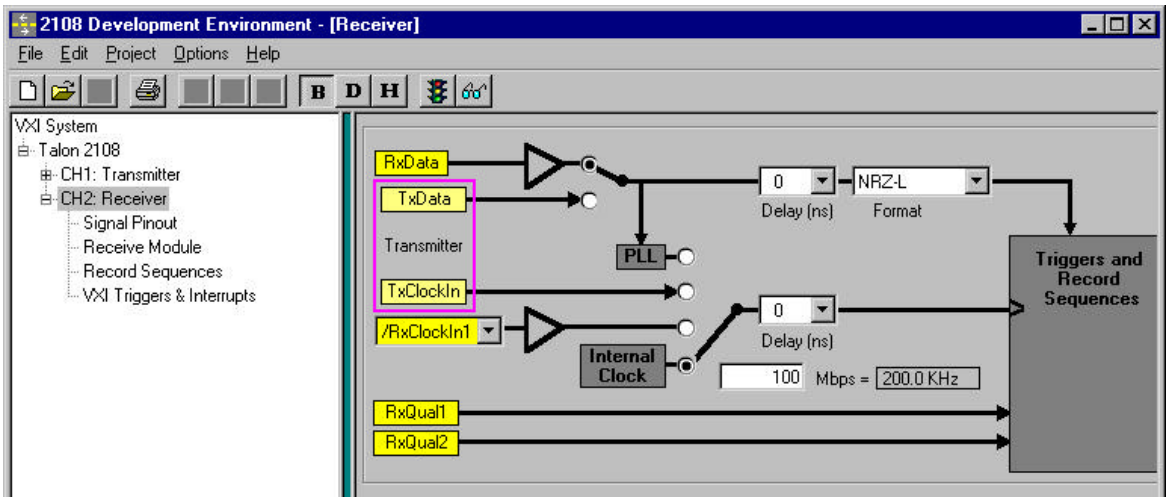
a) Data- if the Receiver clock is to be derived from the data stream then the user has the choice of data coming in via the RXData path from the receiver font panel pins 2A and 4A or if the 2108 has been set to bi-directional mode then the selection is the TxData path which is shared between the 2108RX and the adjacent 2108TX.



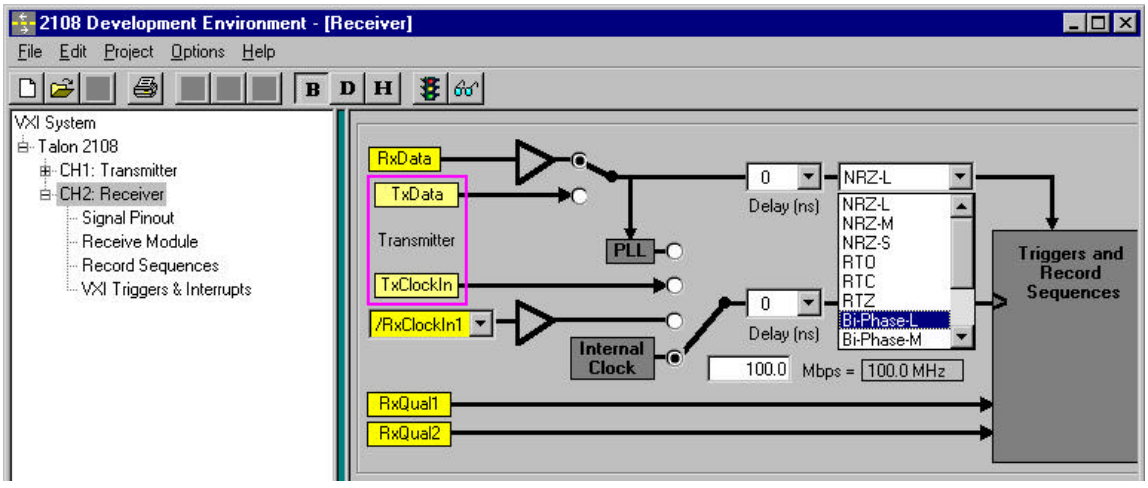
b) External- if the receiver is to use an external clock input then the user has selects the TxClockIn if running in bi-directional mode or one of the receiver external clocks from the pull-down selection.



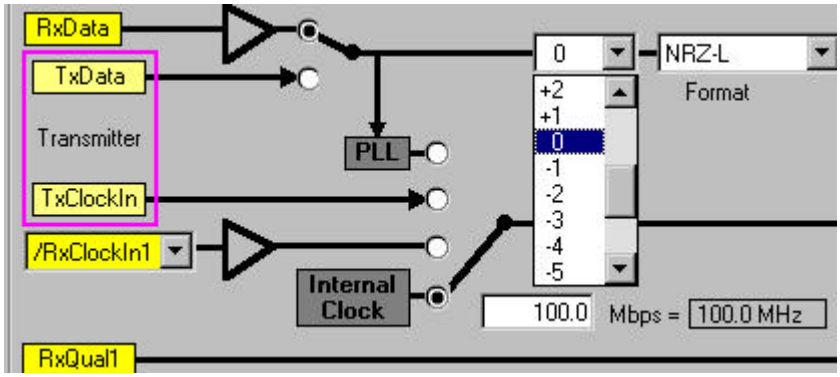
c) Internal- the last option is to use the internal clock which is fully programmable between 2Kbps and 200Mbps. Selecting the Internal Clock connect button will open the entry window.



2) Data Format- the receiver must know the type of data format in order to properly record the incoming data. The data format is selected in the Format using the pull-down menu.

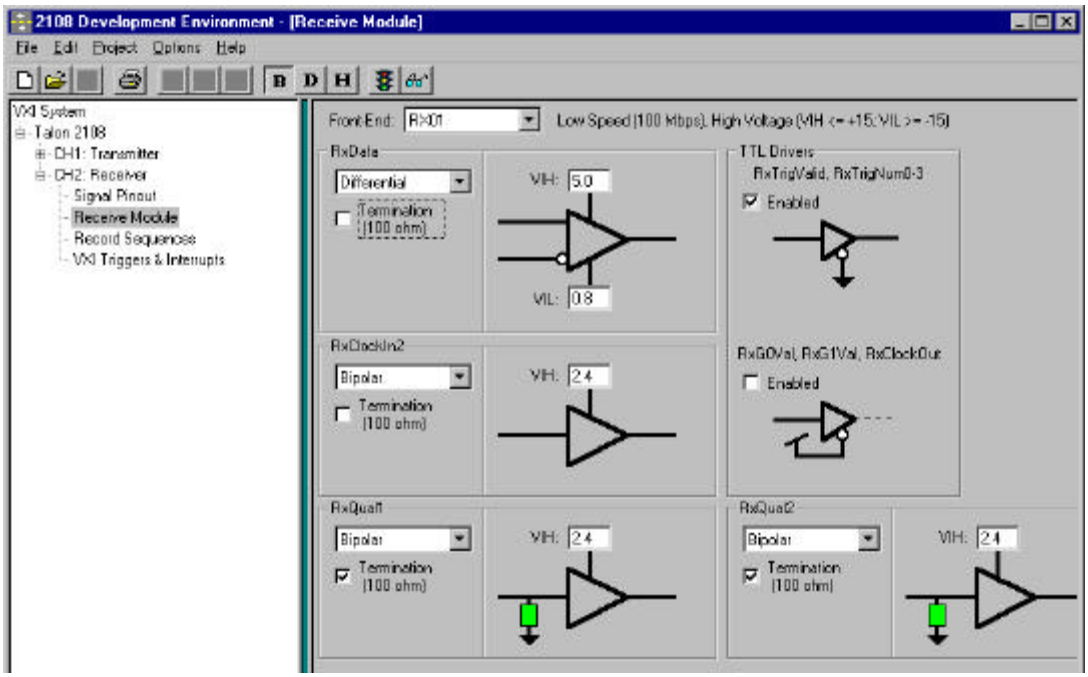


3) Delay- delays in 1ns increments from +/- 10ns may be to the clock and/or the data paths to correct for synchronization problems due to cables, etc.



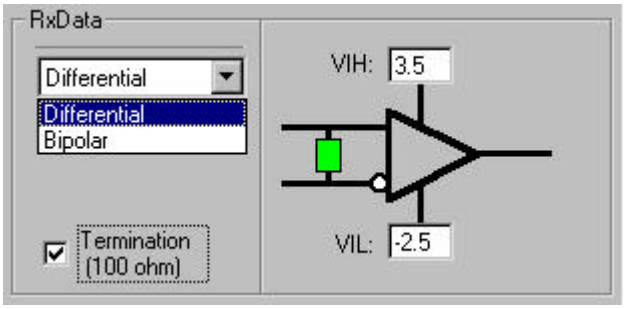
5.2.1.1.2 RECEIVE MODULE PANEL

The physical bit characteristics, data voltage levels, signal types, etc., for the record data are programmed in the Receive Module panel.

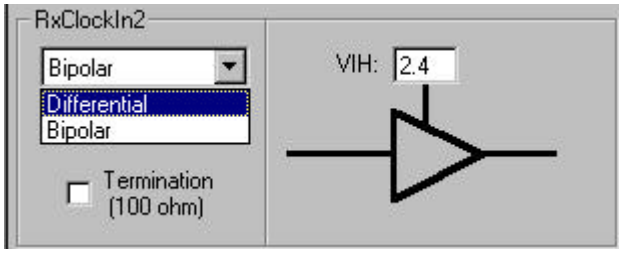


he signals to be programmed are:

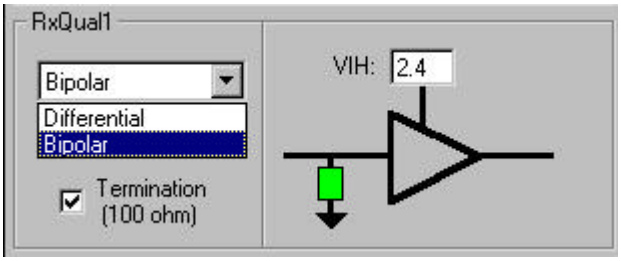
1) RxData- the selections to be programmed for the receive data are the type, Differential or Bi-polar, Termination or none, and the voltage levels VIH and VIL.



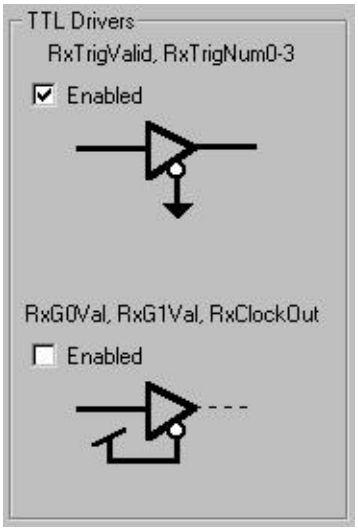
2) RxClockIn2- the receiver external clock input, RxClockIn2, is programmable. The signal type, termination signal and the “Good” level is programmed in this panel.



3) RxQual1 & RxQual2- the 2108RX is capable of being triggered by an external trigger source. RxQual1and RxQual2 are the two qualifier signals and both are programmable. The signal type, termination and “Good” signal level may be programmed.

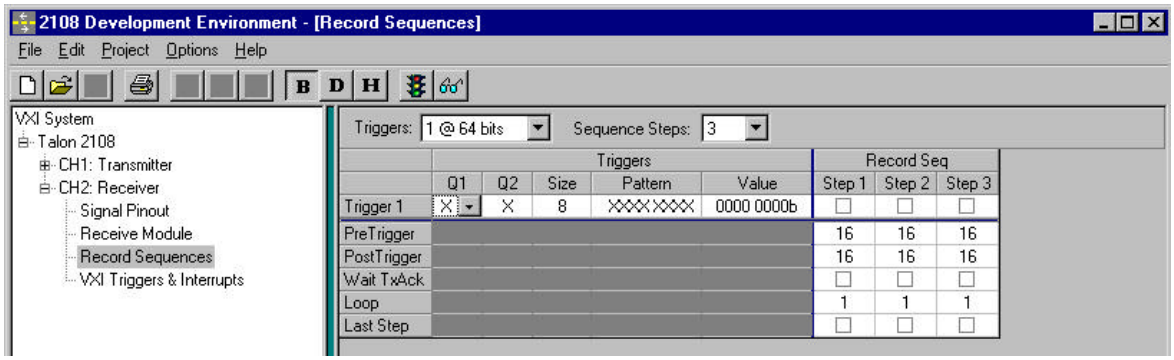


4.) TTL Signal Drivers- the RX01 and RX02 drive the user selectable signals explained in the Signal Pinout section above with TTL drivers. The drivers may be enabled or not according to whether the user is using them in the project. This panel allows the user to set the state of the drivers.



5.3 RECORD SEQUENCES PANEL

The Record Sequences panel is used to define the trigger conditions to initiate the recording of data by the 2108RX. The conditions may be just a single signal going high or more complex scenarios involving data patterns. The following examples using the Record Sequences panel will illustrate the power of the triggering logic of the 2108RX.



5.4 TRIGGERING

The 2108RX incorporates a special purpose TRIGGER SEQUENCER which allows a multitude of trigger combinations. Fig 1 depicts the soft panel which allows the set-up for the TRIGGER SEQUENCE. The TRIGGER SEQUENCE consists of two basic elements: the TRIGGER REGISTERS and the TRIGGER SEQUENCER.

5.5 TRIGGER REGISTERS

As shown in Fig 1, there are 16 thirty-two bit TRIGGER REGISTERS, nomenclated TRIGGER 1 through TRIGGER 16 (alternately, these may be software configured to 8 sixty-four bit trigger registers). The user can program up to 16 different trigger patterns, each pattern being from 1 bit to 32 bits. Any bit within a trigger pattern may be masked off.

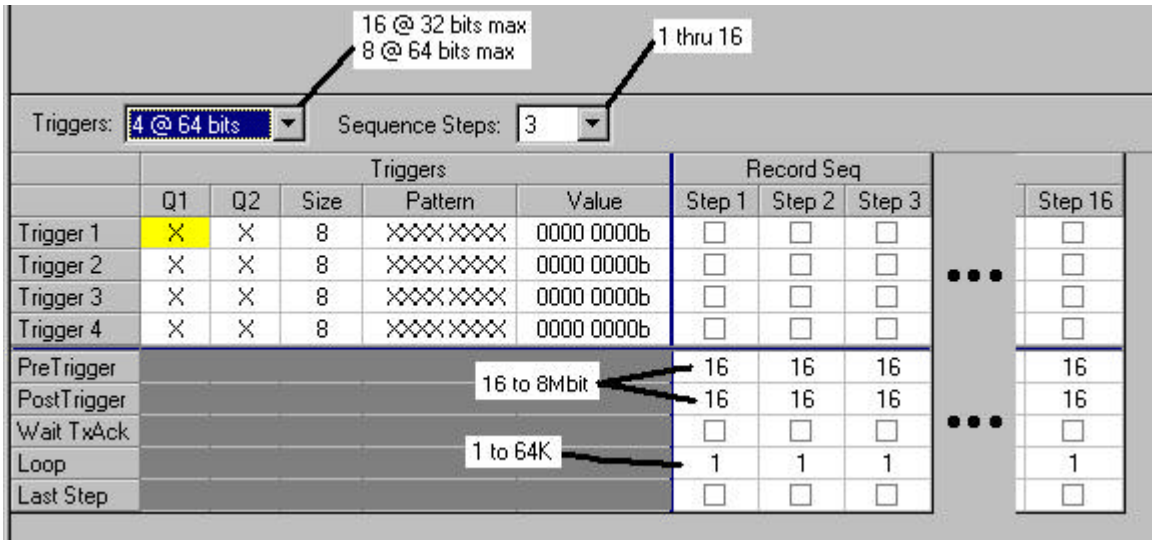


Fig. 1

Once started, the 2108RX is constantly recording data while searching for a trigger pattern. Once triggered, a fixed number of bits prior to the trigger is saved plus a fixed number of bits after the trigger. These bits are referred to as the PRE RECORD data and the POST RECORD data. Fig 1 depicts the minimum and maximum values for the pre and post data. Note the total of all recorded data is 8 megabits.

In addition to the trigger pattern, two QUALIFIER signals may also be added to the trigger equation (reference section 2.6). After the trigger registers are defined, the sequence of the trigger search is entered. This may be best described with a set of examples.

5.6 TRIGGER EXAMPLE #1: TRIGGER AND RECORD

Fig 2 depicts a typical serial data stream.

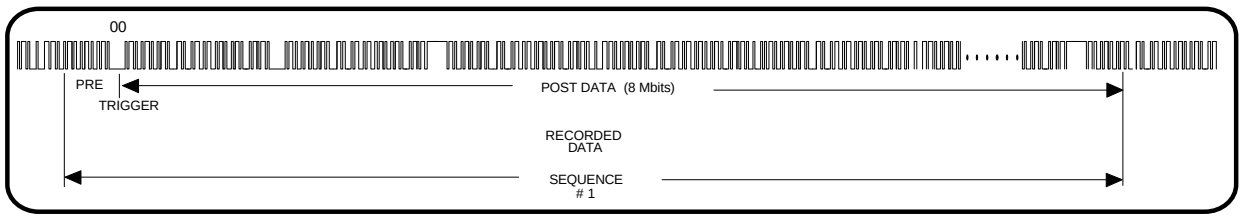


Fig. 2

Assume it is desired to trigger on the first occurrence of the serial hexadecimal character h00. To achieve this, TRIGGER REGISTER 1 would be set to h00, Fig 3. A sequence of 1 STEP would be entered. STEP 1, TRIGGER 1 would then be enabled (shown with a check mark). The PRE RECORD bit length is set to 64 bits; the POST RECORD bit length is set to 8,000,000 bits with a LOOP count of 1.

Once the serial data value of h00 is detected, the 2108RX will save the previous 64 data bits prior to the trigger and then record an additional 8 megabits of data after the trigger, Fig 2.

Triggers: 4 @ 64 bits Sequence Steps: 3

	Triggers					Record Seq		
	Q1	Q2	Size	Pattern	Value	Step 1	Step 2	Step 3
Trigger 1	X	X	8	0000 0000	0x00	☑	☐	☐
Trigger 2	X	X	8	XXXX XXXX	0x00	☐	☐	☐
Trigger 3	X	X	8	XXXX XXXX	0x00	☐	☐	☐
Trigger 4	X	X	8	XXXX XXXX	0x00	☐	☐	☐
PreTrigger						64	16	16
PostTrigger						8000000	16	16
Wait TxAck						☐	☐	☐
Loop						1	1	1
Last Step						☐	☐	☐

Fig. 3

5.7 TRIGGER EXAMPLE #2: SEARCH FOR A SEQUENCE OF DATA VALUES

The following example, although it may have no practical value, should provide the reader with a further understanding of the trigger capability of the 2108RX.

Assume it is desired to record the data surrounding a sequential set of characters. In particular, assume we first want to record 64 bits (32 PRE DATA bits and 32 POST DATA bits) when the hex character h00 is detected. We then want to wait for the hex character h11, then h22 and so forth.
Fig 4 depicts the serial data train and the desired recorded data. Fig 5 depicts the TRIGGER SEQUENCE. STEP 1 would be enabled to search for the hex characters h00 with the LOOP set to the value 1. After the trigger occurs, the sequencer would go to STEP 2, which in turn is enabled to search for the value h11. The process will continue until STEP 16 is executed.

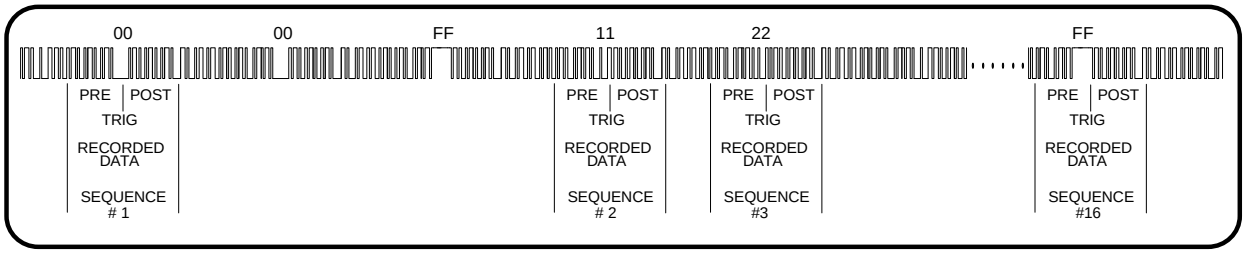


Fig. 4

Triggers: 16 @ 32 bits Sequence Steps: 16

	Triggers					Record Seq									
	Q1	Q2	Size	Pattern	Value	Step 1	Step 2	Step 3	Step 4	Step 5	Step 6	Step 7	Step 8	Step 9	Step 16
Trigger 1	X	X	8	0000 0000	0x00	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 2	X	X	8	0001 0001	0x11	☐	☑	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 3	X	X	8	0010 0010	0x22	☐	☐	☑	☐	☐	☐	☐	☐	☐	☐
Trigger 4	X	X	8	0011 0011	0x33	☐	☐	☐	☑	☐	☐	☐	☐	☐	☐
Trigger 5	X	X	8	0100 0100	0x44	☐	☐	☐	☐	☑	☐	☐	☐	☐	☐
Trigger 6	X	X	8	0101 0101	0x55	☐	☐	☐	☐	☐	☑	☐	☐	☐	☐
Trigger 7	X	X	8	0110 0110	0x66	☐	☐	☐	☐	☐	☐	☑	☐	☐	☐
Trigger 8	X	X	8	0111 0111	0x77	☐	☐	☐	☐	☐	☐	☐	☑	☐	☐
Trigger 9	X	X	8	1000 1000	0x88	☐	☐	☐	☐	☐	☐	☐	☐	☑	☐
Trigger 10	X	X	8	1001 1001	0x99	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 11	X	X	8	1010 1010	0xAA	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 12	X	X	8	1011 1011	0xBB	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 13	X	X	8	1100 1100	0xCC	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 14	X	X	8	1101 1101	0xDD	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 15	X	X	8	1110 1110	0xEE	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 16	X	X	8	1111 1111	0xFF	☐	☐	☐	☐	☐	☐	☐	☐	☐	☑
PreTrigger						32	32	32	32	32	32	32	32	32	32
PostTrigger						32	32	32	32	32	32	32	32	32	32
Wait TxAck						☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Loop						1	1	1	1	1	1	1	1	1	1
Last Step						☐	☐	☐	☐	☐	☐	☐	☐	☐	☐

Fig. 5

5.8 TRIGGER EXAMPLE #3: SEARCH FOR MULTIPLE DATA VALUES

Note the recorded results, Fig 6, if TRIGGER 1 is also enabled in STEP 2 of the SEQUENCER. At STEP 2, the SEQUENCER will now search for the character occurrence of either h00 or h11, Fig 7.

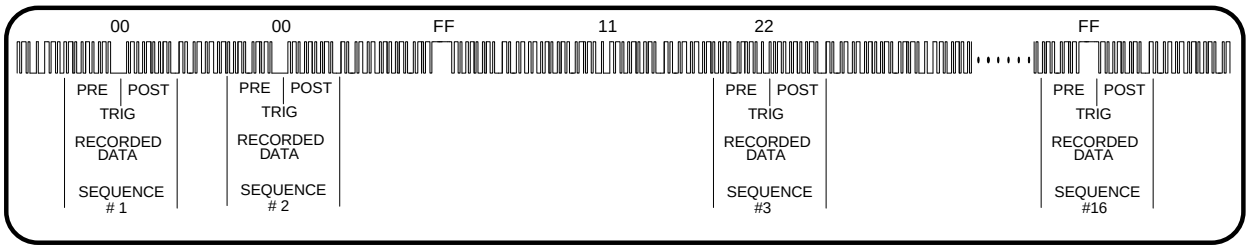


Fig. 6

Triggers: 16 @ 32 bits Sequence Steps: 16

	Triggers					Record Seq									
	Q1	Q2	Size	Pattern	Value	Step 1	Step 2	Step 3	Step 4	Step 5	Step 6	Step 7	Step 8	Step 9	Step 16
Trigger 1	X	X	8	0000 0000	0x00	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 2	X	X	8	0001 0001	0x11	☐	☑	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 3	X	X	8	0010 0010	0x22	☐	☐	☑	☐	☐	☐	☐	☐	☐	☐
Trigger 4	X	X	8	0011 0011	0x33	☐	☐	☐	☑	☐	☐	☐	☐	☐	☐
Trigger 5	X	X	8	0100 0100	0x44	☐	☐	☐	☐	☑	☐	☐	☐	☐	☐
Trigger 6	X	X	8	0101 0101	0x55	☐	☐	☐	☐	☐	☑	☐	☐	☐	☐
Trigger 7	X	X	8	0110 0110	0x66	☐	☐	☐	☐	☐	☐	☑	☐	☐	☐
Trigger 8	X	X	8	0111 0111	0x77	☐	☐	☐	☐	☐	☐	☐	☑	☐	☐
Trigger 9	X	X	8	1000 1000	0x88	☐	☐	☐	☐	☐	☐	☐	☐	☑	☐
Trigger 10	X	X	8	1001 1001	0x99	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 11	X	X	8	1010 1010	0xAA	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 12	X	X	8	1011 1011	0xBB	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 13	X	X	8	1100 1100	0xCC	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 14	X	X	8	1101 1101	0xDD	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 15	X	X	8	1110 1110	0xEE	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 16	X	X	8	1111 1111	0xFF	☐	☐	☐	☐	☐	☐	☐	☐	☐	☑
PreTrigger						32	32	32	32	32	32	32	32	32	32
PostTrigger						32	32	32	32	32	32	32	32	32	32
Wait TxAck						☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Loop						1	1	1	1	1	1	1	1	1	1
Last Step						☐	☐	☐	☐	☐	☐	☐	☐	☐	☐

Fig. 7

If two or more triggers are enabled in a particular step, the 2108RX will search and trigger if any of the trigger values are detected. Fig 8 and fig 9 depict the recorded data and SEQUENCE set up menu when all the triggers are enabled. For STEP 1 note the LOOP count is set to 16 (reference Section 2.5).

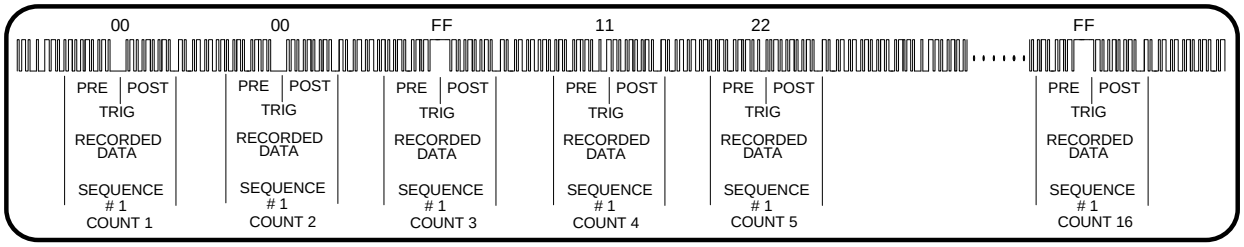


Fig. 8

Triggers: 16 @ 32 bits Sequence Steps: 16

	Triggers					Record Seq									
	Q1	Q2	Size	Pattern	Value	Step 1	Step 2	Step 3	Step 4	Step 5	Step 6	Step 7	Step 8	Step 9	Step 16
Trigger 1	X	X	8	0000 0000	0x00	☑	☑	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 2	X	X	8	0001 0001	0x11	☐	☑	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 3	X	X	8	0010 0010	0x22	☐	☐	☑	☐	☐	☐	☐	☐	☐	☐
Trigger 4	X	X	8	0011 0011	0x33	☐	☐	☐	☑	☐	☐	☐	☐	☐	☐
Trigger 5	X	X	8	0100 0100	0x44	☐	☐	☐	☐	☑	☐	☐	☐	☐	☐
Trigger 6	X	X	8	0101 0101	0x55	☐	☐	☐	☐	☐	☑	☐	☐	☐	☐
Trigger 7	X	X	8	0110 0110	0x66	☐	☐	☐	☐	☐	☐	☑	☐	☐	☐
Trigger 8	X	X	8	0111 0111	0x77	☐	☐	☐	☐	☐	☐	☐	☑	☐	☐
Trigger 9	X	X	8	1000 1000	0x88	☐	☐	☐	☐	☐	☐	☐	☐	☑	☐
Trigger 10	X	X	8	1001 1001	0x99	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 11	X	X	8	1010 1010	0xAA	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 12	X	X	8	1011 1011	0xBB	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 13	X	X	8	1100 1100	0xCC	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 14	X	X	8	1101 1101	0xDD	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 15	X	X	8	1110 1110	0xEE	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 16	X	X	8	1111 1111	0xFF	☐	☐	☐	☐	☐	☐	☐	☐	☐	☑
PreTrigger						32	32	32	32	32	32	32	32	32	32
PostTrigger						32	32	32	32	32	32	32	32	32	32
Wait TxAck						☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Loop						1	1	1	1	1	1	1	1	1	1
Last Step						☐	☐	☐	☐	☐	☐	☐	☐	☐	☐

Fig. 9

5.9 **TRIGGER EXAMPLE #4:** RECORD A MULTIPLE NUMBER OF WORDS

Each SEQUENCE step incorporates a LOOP count value. This enables the recording of multiple words, each with a common or unique trigger value. Assume the code h00 precedes each data word to be captured and we want to trigger and record 1000 data words. Fig 10 and Fig 11 depict this scenario.

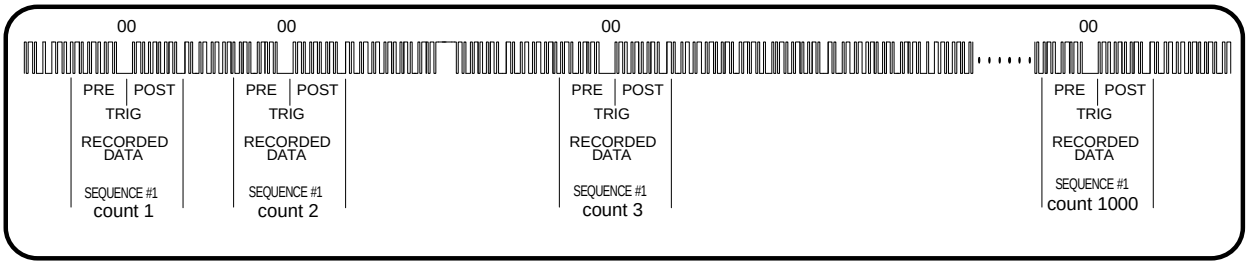


Fig. 10

Triggers: 16 @ 32 bits Sequence Steps: 16

	Triggers					Record Seq									
	Q1	Q2	Size	Pattern	Value	Step 1	Step 2	Step 3	Step 4	Step 5	Step 6	Step 7	Step 8	Step 9	Step 16
Trigger 1	X	X	8	0000 0000	0x00	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 2	X	X	8	0001 0001	0x11	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 3	X	X	8	0010 0010	0x22	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 4	X	X	8	0011 0011	0x33	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 5	X	X	8	0100 0100	0x44	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 6	X	X	8	0101 0101	0x55	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 7	X	X	8	0110 0110	0x66	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 8	X	X	8	0111 0111	0x77	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 9	X	X	8	1000 1000	0x88	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 10	X	X	8	1001 1001	0x99	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 11	X	X	8	1010 1010	0xAA	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 12	X	X	8	1011 1011	0xBB	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 13	X	X	8	1100 1100	0xCC	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 14	X	X	8	1101 1101	0xDD	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 15	X	X	8	1110 1110	0xEE	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 16	X	X	8	1111 1111	0xFF	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐
PreTrigger						32	32	32	32	32	32	32	32	32	32
PostTrigger						32	32	32	32	32	32	32	32	32	32
Wait TxAck						☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Loop						1	1	1	1	1	1	1	1	1	1
Last Step						☐	☐	☐	☐	☐	☐	☐	☐	☐	☐

Fig. 11

5.10 TRIGGER EXAMPLE #5: TRIGGER ON QUALIFIER

In addition to the 16 trigger registers, there are two QUALIFIER signals. These two signals can be logically combined with the trigger registers to form a more complex trigger event.

Assume an interface uses a separate signal, a DATA ENABLE signal, to indicate that the serial data is valid. Fig 12 and Fig 13 depict the recorded data and SEQUENCE set up to record 1000 words, one for each time the DATA ENABLE goes true (Active Low).

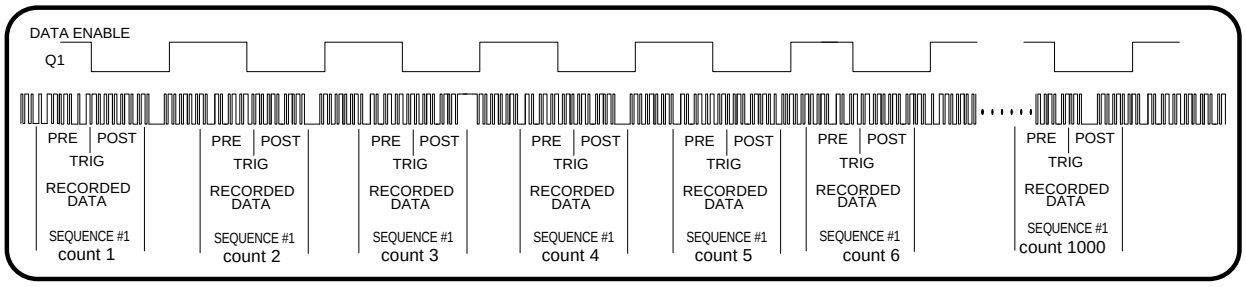


Fig. 12

Triggers: 16 @ 32 bits Sequence Steps: 16

	Triggers					Record Seq									
	Q1	Q2	Size	Pattern	Value	Step 1	Step 2	Step 3	Step 4	Step 5	Step 6	Step 7	Step 8	Step 9	Step 16
Trigger 1	X	X	8	0000 0000	0x00	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 2	X	X	8	0001 0001	0x11	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 3	X	X	8	0010 0010	0x22	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 4	X	X	8	0011 0011	0x33	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 5	X	X	8	0100 0100	0x44	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 6	X	X	8	0101 0101	0x55	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 7	X	X	8	0110 0110	0x66	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 8	X	X	8	0111 0111	0x77	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 9	X	X	8	1000 1000	0x88	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 10	X	X	8	1001 1001	0x99	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 11	X	X	8	1010 1010	0xAA	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 12	X	X	8	1011 1011	0xBB	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 13	X	X	8	1100 1100	0xCC	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 14	X	X	8	1101 1101	0xDD	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 15	X	X	8	1110 1110	0xEE	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 16	X	X	8	1111 1111	0xFF	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
PreTrigger						32	32	32	32	32	32	32	32	32	32
PostTrigger						32	32	32	32	32	32	32	32	32	32
Wait TxAck						☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Loop						1000	1	1	1	1	1	1	1	1	1
Last Step						☐	☐	☐	☐	☐	☐	☐	☐	☐	☐

Fig. 13

The QUALIFIERS can also be logically combined with the trigger registers. Fig 14 and Fig 15 depict the recording of data when the QUALIFIER 1 is true and the data value of hFF is detected. After this occurrence, STEP 2 will record 1000 data words only when DATA ENABLE is true.

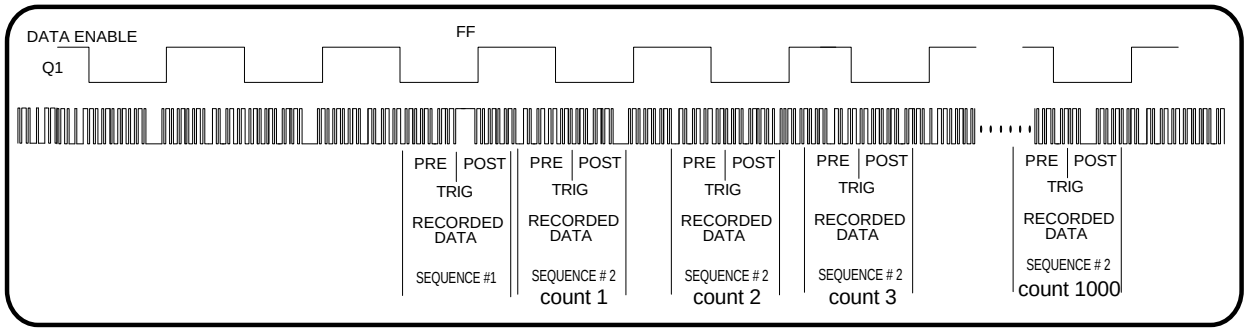


Fig. 14

Triggers: 16 @ 32 bits Sequence Steps: 16

	Triggers					Record Seq									
	Q1	Q2	Size	Pattern	Value	Step 1	Step 2	Step 3	Step 4	Step 5	Step 6	Step 7	Step 8	Step 9	Step 16
Trigger 1	L	X	8	XXXX XXXX	0x00	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 2	X	X	8	0010 0010	0x22	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 3	X	X	8	0010 0010	0x22	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 4	X	X	8	0011 0011	0x33	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 5	X	X	8	0100 0100	0x44	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 6	X	X	8	0101 0101	0x55	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 7	X	X	8	0110 0110	0x66	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 8	X	X	8	0111 0111	0x77	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 9	X	X	8	1000 1000	0x88	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 10	X	X	8	1001 1001	0x99	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 11	X	X	8	1010 1010	0xAA	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 12	X	X	8	1011 1011	0xBB	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 13	X	X	8	1100 1100	0xCC	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 14	X	X	8	1101 1101	0xDD	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 15	X	X	8	1110 1110	0xEE	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 16	X	X	8	1111 1111	0xFF	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
PreTrigger						32	32	32	32	32	32	32	32	32	32
PostTrigger						32	32	32	32	32	32	32	32	32	32
Wait TxAck						☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Loop						1000	1	1	1	1	1	1	1	1	1
Last Step						☐	☐	☐	☐	☐	☐	☐	☐	☐	☐

Fig. 15

5.11 **TRIGGER EXAMPLE #6:** TRIGGER ON WAVEFORM

Each data bit of a serial bus consists of a particular bit format; AMI, BiPhase, NRZ, etc. Once the 2108RX knows the bit format, the 2108RX compensates for the format such that the user enters his trigger data in a user orientated format.

Many serial interfaces use illegal data formats for the purpose of syncing the transmitter with the receiver. The 1553 interface is an example of this. A legal Manchester code does not allow for three consecutive 1/2 bit times of ones followed by three consecutive 1/2 bit times of zeros. The 1553 takes advantage of this by using this illegal code to indicate that data following the illegal code is valid data. It is therefore imperative that the 2108RX TRIGGER logic have the ability to TRIGGER on valid data streams as well as invalid data streams.

The TRIGGER on WAVEFORM function accomplishes this function. In addition to the TRIGGER and QUALIFIER data, the user may enter a WAVEFORM in the TRIGGER "equation". The WAVEFORM is always entered in 1/2 bit times. Fig 16 and Fig 17 depict the trigger set up for a waveform combined with a data value.

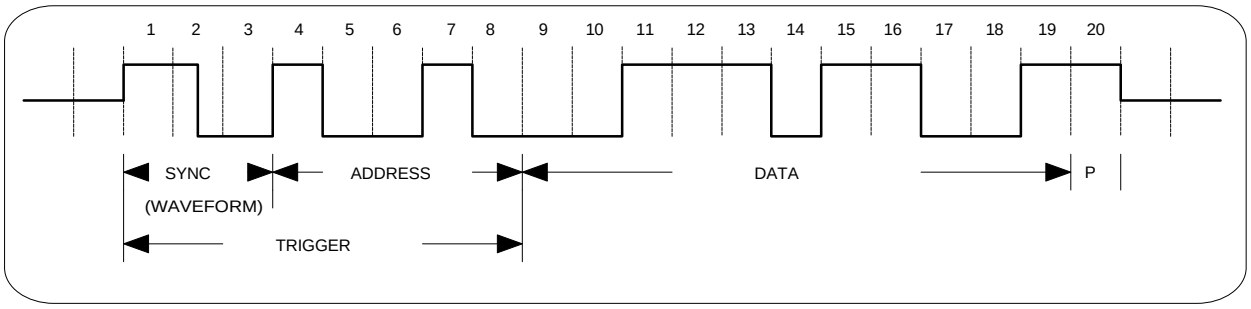


Fig. 16

Triggers: 16 @ 32 bits Sequence Steps: 16

	Triggers					Record Seq									
	Q1	Q2	Size	Pattern	Value	Step 1	Step 2	Step 3	Step 4	Step 5	Step 6	Step 7	Step 8	Step 9	Step 16
Trigger 1	L	X	8	1111 1111	0xFF	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 2	X	X	8	XXXX XXXX	0x00	☐	☑	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 3	X	X	8	0010 0010	0x22	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 4	X	X	8	0011 0011	0x33	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 5	X	X	8	0100 0100	0x44	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 6	X	X	8	0101 0101	0x55	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 7	X	X	8	0110 0110	0x66	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 8	X	X	8	0111 0111	0x77	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 9	X	X	8	1000 1000	0x88	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 10	X	X	8	1001 1001	0x99	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 11	X	X	8	1010 1010	0xAA	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 12	X	X	8	1011 1011	0xBB	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 13	X	X	8	1100 1100	0xCC	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 14	X	X	8	1101 1101	0xDD	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 15	X	X	8	1110 1110	0xEE	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Trigger 16	X	X	8	1111 1111	0xFF	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
PreTrigger						32	32	32	32	32	32	32	32	32	32
PostTrigger						32	32	32	32	32	32	32	32	32	32
Wait TxAck						☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Loop						1	1000	1	1	1	1	1	1	1	1
Last Step						☐	☐	☐	☐	☐	☐	☐	☐	☐	☐

Fig. 17