



HEWLETT  
PACKARD

# **HP 81200 System Getting Started Guide**



---

# HP 81200 System Getting Started Guide

Edition 1.1 related to HP E4873A Software Version 1.10 or higher

Boeblingen Verification Solutions

## Legal Notice

This document contains proprietary information that is protected by copyright. All rights are reserved.

No part of this document may be photocopied, reproduced, or translated to another language without the prior written consent of Hewlett-Packard GmbH.

© Copyright 1998 by:  
Hewlett-Packard GmbH  
Herrenberger Str. 130  
71034 Böblingen  
Germany

---

## Subject Matter

The information in this document is subject to change without notice.

Hewlett-Packard makes no warranty of any kind with regard to this printed material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose.

Hewlett-Packard shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this material.

## ISO 9001 Certification

Produced to ISO 9001 international quality system standard as part of our objective of continually increasing customer satisfaction through improved process control.

## Printing History

New editions are complete revisions of the guide reflecting alterations in the functionality of the instrument. Updates are occasionally made to the guide between editions. The date on the title page changes when an updated guide is published. To find out the current edition of the guide, see above. To purchase an updated guide, contact your Hewlett-Packard representative.

Literature Number: E4849-91012

You could also check the following Internet address to find out, or to download the latest edition:

<http://www.hp.com/go/dvt>

## Trademarks

Windows NT 4.0 is registered trademark of Microsoft Corp. in US and other countries

Pentium is registered trademark of Intel Corp. in US and other countries

Acrobat Reader is registered trademark of Adobe Corp. in US and other countries.

---



<b>Preface .....</b>	<b>7</b>
About this Guide .....	7
<b>Chapter 1 Product and Concepts Overview .....</b>	<b>9</b>
<b>What you can do with the HP 81200 System .....</b>	<b>10</b>
Verify and characterize digital devices .....	10
Emulate real pattern and waveform conditions .....	10
Testing up to 660 Mbit/s on a Single Output/Input Connector .....	10
Testing up to 1.3 Gbit/s with Two 660 Mbit/s Outputs in Channel	
Add Mode .....	11
Create data streams in the lengths as required in your application .....	11
Zero Adjust, Cable and Propagation Delay Compensation .....	12
Negative Delay .....	12
Configure the hardware resources (generator and analyzer channels)	
as required .....	12
Remote Controlled Operation .....	12
Functional tests .....	12
Error analysis and margin tests .....	13
<b>Components of the HP 81200 System .....</b>	<b>14</b>
Modules .....	14
Front-ends .....	15
Configurations .....	16
<b>Using the HP 81200 System .....</b>	<b>17</b>
<b>Data Generation and Analysis .....</b>	<b>18</b>
Naming Conventions .....	18
Segments .....	18
Sequences .....	18
<b>Sequence Generation .....</b>	<b>20</b>
Data Memory Usage .....	20
Segments and Loops .....	20
Segment Type Combinations .....	21
Data to Connector assignment .....	21
<b>External Clock and External Input .....</b>	<b>22</b>
The available Modes .....	22
<b>Software Structure .....</b>	<b>25</b>

	<b>Hardware Resources .....</b>	<b>27</b>
	Connectors .....	27
	Labeling .....	27
	Channels .....	28
<b>Chapter 2</b>	<b>Setting Up Concepts .....</b>	<b>29</b>
	<b>Introduction of the Example .....</b>	<b>30</b>
	Key Points for DUT Scheme Modeling and Signal Setup .....	30
	Key Points for Verification .....	31
	Initial Setup .....	31
	Test Setup .....	31
	<b>Create Model of Device Under Test (DUT) .....</b>	<b>33</b>
	Opening the Connection Window .....	35
	Creating the various ports .....	35
	Renaming the terminals. ....	36
	<b>Associate Resources of HP 81200 System with DUT .....</b>	<b>38</b>
	<b>Set System Parameters .....</b>	<b>40</b>
	<b>Set Signal Parameters .....</b>	<b>41</b>
	<b>Save Settings .....</b>	<b>44</b>
	<b>Create the Sequence .....</b>	<b>46</b>
	<b>Import the Data Segments .....</b>	<b>49</b>
	<b>RunTest .....</b>	<b>52</b>
	<b>Comparison of Actual and Theoretical Output Waveforms .....</b>	<b>53</b>
<b>Chapter 3</b>	<b>Controlling the HP 81200 using HP VEE .....</b>	<b>55</b>
	<b>A simple Example using VEE .....</b>	<b>56</b>
	General Information .....	56
	Hardware Resources .....	56
	Steps to do for Remote Programming the HP 81200 System by HP VEE via HP-IB .....	57
	Test Setup .....	57
	Sample Program .....	57
	Procedure .....	57

---

# Preface

## About this Guide

This guide gives product and concepts overview and an application independent example of how to use the HP 81200 System from the graphical user interface (GUI).

The example is designed so that you can go through step by step to learn the concepts and methods to setup the HP 81200 System to create the required signals. Each task is explained independently of the others allowing you to use the guide also as a quick reference.





---

# Chapter 1    **Product and Concepts Overview**

---

This chapter provides a brief overview of the HP 81200 System, its software and some key concepts.

The following are described:

- *“What you can do with the HP 81200 System” on page 10*
- *“Components of the HP 81200 System” on page 14*
- *“Using the HP 81200 System” on page 17*
- *“Data Generation and Analysis” on page 18*
- *“Sequence Generation” on page 20*
- *“External Clock and External Input” on page 22*
- *“Software Structure” on page 25*
- *“Hardware Resources” on page 27*

## What you can do with the HP 81200 System

The HP 81200 System is a multi-purpose digital stimulus/response system. This section provides an overview of what can be achieved using this system. The HP 81200 System can be operated manually on a bench top or remote controlled in an automated test rack. The system may be controlled over HP-IB or a LAN using command strings.

### Verify and characterize digital devices

The Device Under Test (DUT) and its application setup is modeled in software. In the Graphical User Interface (GUI) the raw DUT scheme is shown in the Connection Window. In the Connection Window it is possible to group signals required for the DUT, these groups of signals are called Ports. There are two major port types possible, either data signals with variable pulse parameters (DATA Ports) or pure parametric signals (PULSE Ports), also used for clock signals. All signal parameters may be set up for individual signals for DUT input connector (input terminal) or a group of terminals (a port). The system has data generation and analyzing frontends. Cable Delays and signal skew in the test setup may be compensated by using the deskew feature (See [Chapter 8 “Deskew / Delay Compensation” on page 95](#) of the Installation Guide and also [“Deskew Editor” on page 97](#) of the Reference Guide).

### Emulate real pattern and waveform conditions

Data patterns (data segments) may be stored in the system database and output as part of a sequence with algorithmic data such as a PRBS. Data Patterns for the signals required by the DUT can easily be set up in terms of data segments which span across several output or input connectors of the HP 81200 System. Captured data or data produced by a simulation may be imported as an (ASCII) text file.

The HP 81200 System can be used to stimulate communications devices using its sequencing capability. Packets or cells consisting of payload and control data may be produced by creating control segments and using a PRBS segment for the payload. Cell/packet size can be varied and control segments can be stored in the database and used in any number of different packets. A PRBS pattern may be used as the payload to test error rates. Intermittent data with long dead-times between bursts can easily be produced using the pause segment.

For multiplexer/demultiplexer testing it is possible to set up PRWS data and compare segments. Also, it is possible to let different ports run at different frequencies.

### Testing up to 660 Mbit/s on a Single Output/Input Connector

Depending on the front-end it is possible to test devices at frequencies up to 660 MHz. This is achieved by multiplexing. The 1 Mbit memory of each channel is arranged in 64k x 16 bit. With the highest multiplexing factor of 16 it is possible to have up to 660 MHz signals with 1 Mbit maximum memory depth.

The lengths of the segments have a granularity which depends on the system frequency multiplier range (FMR) factor which defines the system clock rate. For information on setting the system clock frequency and system FMR factor see [“Parameter Editor” on page 55](#). in the reference guide. The FMR Factor, see [Table 1 on page 11](#) shows the relationship between segment length resolution, memory depth and maximum system clock rate.

**Table 1** Matrix of Block Length Granularity, Frequency Multiplier Range, Memory Depth and System Clock Frequency

Block Length Granularity	Frequency Multiplier Range <sup>1,2</sup>	Memory Depth <sup>3</sup>	System Clock Rates
1 bit (=1)	1, 2, 4, 8, 16	64 kbit	</= 41.67 MHz
2 bits (=2)	1/2, 1, 2, 8	128 kbit	</= 83.83 MHz
4 bits (=4)	1/4, 1/2, 1, 2, 4	256 kbit	</= 166.67 MHz
8 bits (=8) <sup>4</sup>	1/8, 1/4, 1/2, 1, 2	512 kbit	</= 333.33 MHz
16 bits (=16) <sup>5,6</sup>	1/16, 1/8, 1/4, 1/2, 1	1 Mbit	</= 660 MHz

1 For frequencies below 333.334 kHz it is not possible to select fractions of the frequency for individual connectors. Therefore it is recommended to set the global frequency/period to the lowest/longest frequency/period required and multiply by 2, 4, 8 or 16 at the connectors individual.

2 This is the range of multiples and fractions which can be used at individual connectors. When you have most of your signals at 40 MHz and your pattern lengths are less then 64 kbit, then you can choose block length granularity 1. You have the chance to set individual connectors to a multiple of this general setting, for example selecting 16 as the multiply factor for a connector gives you 1 Mbit memory depth and 640 MHz with a block length granularity of 16.

3 Subtract 32xBlocklength Granularity, as this memory space is occupied by a  $2^5-1$  PRxS and the sequencing initialization.

4 Not available for dual analyzer front-ends in the Error Capture mode.

5 Not available for dual generator and analyzer front-ends.

6 For single-ended analyzer front-ends not available in Error Capture mode.

Each channel/connector can individually be set to multiples or fractions of 2 of the system clock frequency, selected. If, for example most of your signals are at 200 MHz, then the corresponding block length granularity (BLG) you can choose from is either 8 or 16 (block length resolution). If you have chosen 8 as the general BLG, then each data port and pulse port or each terminal of a pulse port can individually be set to frequencies of 1/8, 1/4, 1/2, 1 or 2 times the system clock frequency. When the frequency multiply factor is changed for individual ports or terminals (connectors), then the BLG, memory depth and frequency changes for this connector, for example you multiply the system frequency at a certain data port by 2, then the BLG for this connector is 16, the memory depth is 1 Mbit and the data rate for this port is 400 Mbit/s.

### Testing up to 1.3 Gbit/s with Two 660 Mbit/s Outputs in Channel Add Mode

Two 660 Mbit/s outputs in RZ mode, 50% Duty Cycle and the second output delayed with 50% of the period can be added to achieve a 1.3 Gbit/s NRZ data stream.

### Create data streams in the lengths as required in your application

The available memory depth per channel is up to  $(64\text{ k} - 32) \times 16$  bit. A  $(2^5-1)$  PRxS is stored as default in the memory, covering 31 memory locations. One memory location is required for sequencing initialization. When the system FMR factor 1 is selected, then the available memory depth is  $(64\text{ k} - 32)$ bits. When the system FMR factor 16 is selected, then the available memory depth is  $(64\text{ k} - 32) \times 16$  bit. Memory is saved as only none-repetitive data pattern are stored and not the complete data stream bit by bit. The looping capability with up to 5 loop levels help to create long and complex

data streams which result in real data sequences which are much longer than the available physical channel memory . Data pattern are stored as data segments in the database, when the data segments are not too specific for a certain setting, then these data segments can be handled as global segments, and used across several settings, which helps to save hard disk space dramatically.

## Zero Adjust, Cable and Propagation Delay Compensation

With the HP 81200 System it is possible to synchronize the outputs and inputs of new installed front-ends or new modules. To assure that all generator output signals are applied at the same time either at the DUT-board or even at the DUT input pins it is possible to perform a cable delay and propagation delay compensation from the Go>Deskew Editor window. The procedures are menu driven, please refer to the chapter "[Deskew / Delay Compensation](#)" on page 95 of the Installation Guide.

## Negative Delay

Apart from synchronizing signals it might be interesting and important to have some signals applied in advance to other signals. Therefore the HP 81200 System offers to set a general delay offset for all connectors, so that individual ones can be set to negative delays, so start earlier than others. The delay offset feature can be used in Setup and Hold Time measurements.

## Configure the hardware resources (generator and analyzer channels) as required

The concept of the HP 81200 System is to create out of the actual available hardware resources (generator and analyzer channels) of the system application specific so called virtual instruments. By editing the dvtsys.txt configuration file new virtual instruments can be created. New modules added to the HP 81200 System are automatically added to the dvtsys.txt configuration file.

## Remote Controlled Operation

In remote control operation each virtual instrument needs an individual handle (<Handle>) so that the system knows which virtual instrument should receive the command for executing it. All commands and queries have to be preceded with the virtual instrument's specific handle (<Handle>).

As the software structure is a client-server architecture to allow different operating modes, for example remote control via HP-IB, programming via HP VEE or C/C++, and operating via the graphical user interface (GUI) there is one transport layer and therefore one command language for all these interfaces.

## Functional tests

There are three different measurement modes available. The Capture Data mode, which captures data of interest until the memory is filled you can view the result in a state list. There are two real time compare modes for which expected data segments can be edited, the two modes are the Error Rate Measurement mode and the Compare and Acquire around Error mode. The Error Rate Measurement mode scans the receiving data and shows nearly at real time the resulting actual and accumulated number of bits, the actual and accumulated number of errors, and the actual bit error rate. The Compare and Acquire around Error mode scans data as long as an error occurs, then acquires data when an error occurs. It is possible to define when the system should stop after the occurrence of the error, the range to stop is from 608 up to 64 k bits after an error occurs. The received data with the errors can be viewed as a state error list. It is possible to load expected data segments which may have

been captured from a reference device or imported from a simulation. Each analyzer input has an up to 1Mbit memory depth.

Real time compare up to 330 MHz can be achieved with single-ended analyzer front-ends. Real time compare up to 165 MHz can be achieved with dual-input analyzer front-ends.

### **Error analysis and margin tests**

A device may be stimulated with marginal input signals using the variable pulse parameters provided by the HP 81200 System. Parameters include levels, delay and width and may be varied independently for each channel or for a DUT port as a whole. This allows for level or delay losses in the test setup or test board to be compensated for.

Glitches and pulse delay variation may be emulated using digital addition of two channels. Up to 4 channels can be added to emulate a realtime pulse delay variation with up to 4 phases.

## Components of the HP 81200 System

This section describes the HP 81200 System hardware components.

### Modules

#### Clock Source Modules

The clock module generates the system clock and synchronizes all data generator and analyzer channels in a mainframe. The clock module provides the sequencing capability of a system and may use its internal synthesized clock source or an external clock source. The internal clock synthesis can be synchronized to a common frequency standard using the PLL reference input.

The following clock source modules are available:

- HP E4805A Central Clock Module.  
This module synchronizes up to 11 analyzer and generator modules. It may connect up to two clock modules located in expander frames. This central clock module permits to use up to 5 loop levels.
- HP E4831A Central Clock and Data Generator Module  
This module combines some features of the central clock module and of the data generator module. This module synchronizes up to 6 generator modules. It permits to use up to 2 loop levels.

#### Data Generator/Analyzer Modules

Data signals are either generated by output front-ends, or output signals from the DUT are applied to the analyzer inputs for analysis. The front-ends are installed in the Data Generator/Analyzer module. A Data Generator/Analyzer module has 4 slots for front-ends. It is recommended to load the modules as homogeneously as possible, this means load output front-end in one module and input front-ends in another module.

The following modules are available:

- HP E4841A Data Generator/Analyzer Module:  
This module provides space for four front-ends, with one or two channels each. Any combination of input and output front-ends within a module is possible, but not recommended, as it may conflict with the concept of porting for output and input signals.
- HP E4831A Central Clock and Data Generator Module:  
This module provides space for two generator front-ends with one or two channels each.

## Front-ends

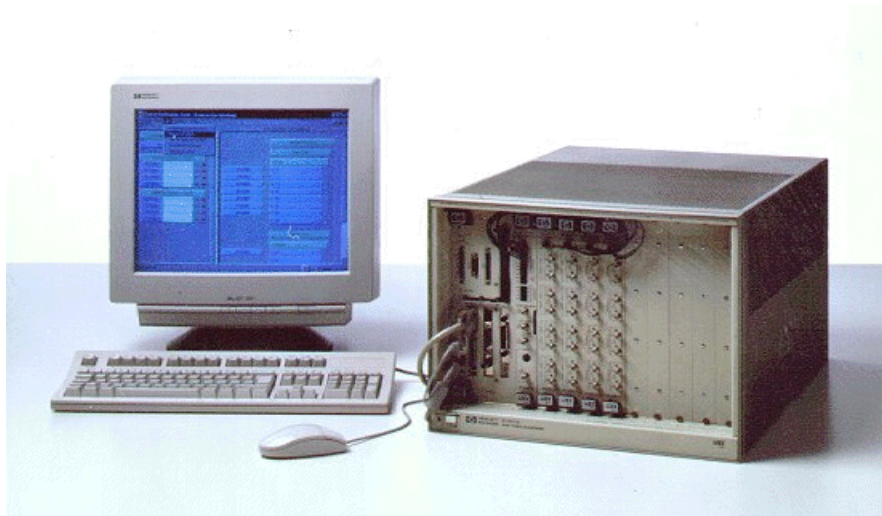
The following front-ends are available:

- HP E4842A 330 Mbit/s, RZ, single ended, variable transitions, 3.5 V amplitude
- HP E4843A 660 MHz Differential Output front-end
- HP E4844A 660 MSa/s Single Input front-end
- HP E4845A 330 MSa/s Dual Input front-end
- HP E4846A 200 Mbit/s Dual Output front-end
- HP E4847A 330 MSa/s Hi-Z Dual Input front-end

## Configurations

The HP 81200 System is available in several configurations. A system may consist of a 3-slot or 13-slot mainframe or up to three 13-slot mainframes. A system may have an integrated display and data entry unit or separate monitor, keyboard and mouse. The system may be operated from the graphical user interface or can be controlled via HP-IB. It has also a programming interface to HP VEE or C/C++.

**Figure 1**                    **13-Slot Mainframe Configuration**



See also [Appendix A “System Specifications” on page 105](#) of the Installation Guide for a detailed description of the various different configurations.



## Using the HP 81200 System

When the HP 81200 System is started, the software builds a scheme of the available hardware (modules and front ends) of the selected 'virtual' instrument, e.g.: 'virtual' instrument DSRA (in remote control DSRA is the handle for this 'virtual' instrument). The HP 81200 System is used by creating a model of your device under test (DUT) in the software and associating the physical resources of the instrument with this 'virtual' DUT. The connections between the terminals of the virtual DUT and the virtual instrument correspond exactly to the physical connections between the DUT and the HP 81200 System.

The various names are defined as follows:

- A **connector** represents a output or input connector on the module. A channel represents the circuitry behind a connector.
- A **terminal** represents a signal at the DUT.
- A **port** is a collection of terminals.

The terminals of a DUT may be grouped into busses/ports, such as data busses/ports, address bus/port and control bus/port.

A virtual DUT is constructed from a scheme (or template). A scheme provides several ports. The HP 1200 System includes a general DUT Scheme.

The general DUT Scheme provides two types of ports:

- **Data Port.** This provides the capability to define data to be sent/analysed. Data is handled by segments. Two major data segment types are available, memory based or PRBS/PRWS. Sequences of segments can be repeated. Individual pulse parameter settings are also possible.
- **Pulse Port.** Like traditional pulse generators, this port provides an easy way to have a pulse generated without the need to setup any data. Also, clock signals or dc signals can be set up.

Ports are added for each group of signals with the same behaviour or which can be treated in the same way in the DUT.

The real physical DUT has to be modelled in the GUI as a virtual DUT. When the virtual DUT has been modeled and has been connected to the virtual instrument, signal parameters can be set, such as signal timings, pulse delay and pulse width. Signal level parameters may also be set up. Signal parameters may be set up globally for each port or individually for each terminal. More complex signals may be produced by digital addition of two or four output channels. This allows to set up real-world-signals like pulse displacement or width variation.

The complete settings for a virtual DUT may be saved in the system's database. It is possible to import a setting as a text file, either in the GUI, or in remote control with the command `:dvt:mmem:sett:imp <hdl>,<"filename">|<(expression)>`.

When the signal parameters have been set up, it is time to apply patterns. The type of pattern that may be applied to a port depends on the type of port. Data ports can output sequences of segments. These sequences can contain loops and may be externally started, stopped or gated. Segments may be stored in a global segment pool or in the segment pool of the current setting.

## Data Generation and Analysis

The segment editor and sequence editor define the data contents and structure of the data streams sent or received by the HP 81200 System. A sequence specifies which data segments are sent (or expected), on which ports and in which timely order. A sequence consists of blocks which may be repeated a specified number of times. Blocks span across all data ports and contain a data segment for each port.

### Naming Conventions

segment	smallest information unit that can be stored.
block	a sequence consists of blocks. Each block spans across all data ports and hold the port specific segment(s).
sequence	the overall data stream consisting of all blocks, segments and repetitions (loops).

### Segments

A segment is an atomic storage element. New segments are created in the Segment Editor (see [“Segment Editor” on page 75](#) of the Reference Guide) or can be imported as vectors in text file (see [“Vector Import and Export Tool” on page 105](#) of the Reference Guide).

A segment has a width, a length (if it is a memory segment) and can hold either algorithmic parameters (PRxS) or free programmable data pattern.

### Storage Areas

Segments are stored in a segment pool, which is part of the system database.

There is one segment pool with global scope and one segment pool with local scope per setting.

- Segments in the local segment pools can only be accessed through its setting.
- Segments in the global segment pool can be accessed from any setting.

### Sequences

A sequence specifies which data segments are sent or expected. See the next section [“Sequence Generation” on page 20](#) for further details. For creating sequences, see [“Sequence Editor” on page 67](#) of the Reference Guide).

A sequence consists of blocks, within which segments are defined. Ports and loops are also part of a sequence. Sequences are independent of data. This is achieved by defining data in the Segment Editor or externally as vectors in a text file and using these segments in the Sequence Editor.

When a new segment was saved to, or has been imported into the segment pool, it may be downloaded into a sequence.

#### NOTE

Signals from pulse-port will not be shown in the sequence model, as there is no data to edit.

### Differences in Looping Memory Type or PRxS Type Segments

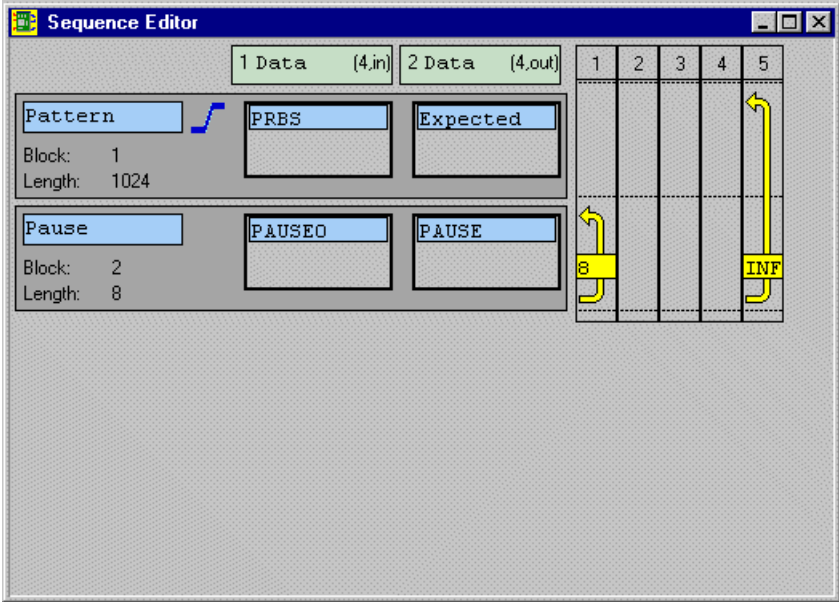
If you use do not use the complete PRxS you have chosen for your application, but are looping it, then with each new loop cycle the next portion of the PRxS is used, when the looping is going on then

happens that the next portion is the rest of the the PRxS and a bit of the beginning. The portoning of the PRxS will go on as long as the looping lasts.

With memory type segments it is different, with each loop the memory type segment is started from the beginning. If the segment is larger then the portion used in the sequence, then there are data which are never generated in this sequence.

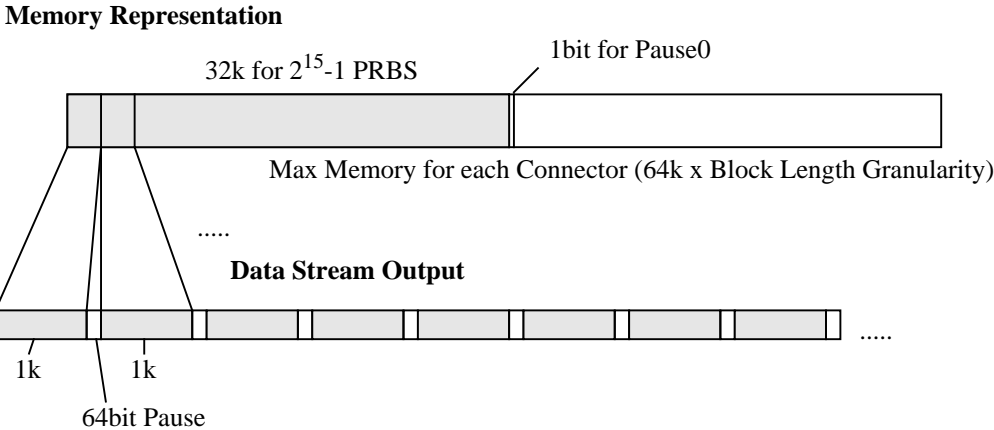
For example, the sequence to loop infinite 1 kbit portions of a  $2^{15}-1$  PRBS and a pause of 64 bits might look like as follows:

**Figure 2** Example Sequence



The data representation in the HP 81200 System’s memory and the data stream output for one connector may look like as follows:

**Figure 3** PRBS Memory Representation and Data Stream Output



## Sequence Generation

Depending on the clock module the sequence capabilities are different. The HP E4831A clock module has 2 loop levels the HP E4805A clock module has 5 loop levels. In the following sections the differences will be explained as required.

### Data Memory Usage

To understand data memory consumption in sequence mode it is best to think in “Words”. A Word consists of 1 ... 16 Bits depending on the Block Length Granularity (Mux-factor).

- 1 Word is reserved for internal used.
- A PRBS/PRWS consumes as many Words as its Polynomial says, e.g. a  $2^{15}-1$  PRBS consumes 32767 Words.
- Even if the user didn't use a PRBS there is a  $2^5-1$  PRBS allocated internally, which means 31 Words are allocated.
- A “Memory Saving Segment” (Pause0/1, ...) consumes 1 Word, if such a segment is used at each channel of the module.
- The remaining memory is used for the data segments.

### Segments and Loops

The number of segments and loops, that can be used depends on the sequencer used.

#### HP E4831A Clock Module:

For the E4831A Clock Module we have the following capabilities:

- 1 Counted Loop-Level with up to 30 counted loops altogether. A counted loop may count up to  $2^{20}$ .
- 1 Infinite Loop in loop-level 2
- Up to 60 Segments: #Segments + #Counted Loops  $\leq$  60. Example: 60 Segments can be used if no segment / sequence of segments is “counted looped”. Up to 30 Segments can each be looped. With 45 Segments up to 15 counted loops can be used (for example always loop 3 Segments).
- 30 Segments minimum available.
- Minimum Segment Length 1 Word, if a counted loop starts at that segment: 2 Words.

#### HP E4805A Clock Module:

For the E4805A Clock Module we have the following capabilities:

- 4 Counted loop levels (1 done on the data modules, 3 done on the clock module). Loop counts may be up to  $2^{20}$ . Note that for loop-level 1 there is a restriction:  $\text{LoopCount1} * \text{SegmentlengthInWords} \leq 2^{20}$ . If this restriction is not met, the loop level 1 can not be used. Instead the looping has to be done on a higher loop-level in this case.
- 1 Infinite Loop in loop-level 5

- Up to 60 Segments.
- 14 Segments minimum available. To overcome this restriction, internally segments are combined if possible. Not only segments are combined, even a loop-level 1 - group may be combined with a following segment or loop-level group. This group must be shorter than  $2^{20}$  unfolded words.
- Minimum Segment Length: 3 Words, this increases to up to 5 Words, if 4 nested counted loops are started. Note that this restriction is sometimes overcome internally by combining a segment with a following segment.

## Segment Type Combinations

- At the same time a generator/ analyzer module can execute either memory type of segments or PRBS/PRWS type of segments only.
- Pause0/Pause1, Expected0/Expected1, Don'tCare segments save memory if at the same time all channels of the module execute such type of segments.
- On a module with analyzer front-ends all connectors can either be in "PAUSE" mode or they must all be in an other mode than "PAUSE", e.g.: capture or acquire.

## Data to Connector assignment

Here the algorithm is described, how the available segment data is assigned to the connectors. If no channel Add is involved, this is done as follows:

- The first terminal within a port gets trace0 , the second gets trace1 and so on. The assignment to the physical connectors depend on what connections you have selected from the terminals to the connectors in the Connection Window in the GUI, and therefore what real connections you make in your application setup.
- If a terminal is connected to a connector where a channel add is done, the connector that holds the connection gets the first trace. An added channel gets the next trace. In this case an exception to the rule "from top to bottom" is made. Added channels are assigned from bottom to top.

## External Clock and External Input

The HP 81200 System can be synchronized to an external clock. Additionally, the system can be started, stopped or gated by an external signal. The external clock mode is selected in the clock reference tab, the start, stop, gated mode is set in the external input tab.

### The available Modes

Figure 4 External Clock Source Selected in the Clock Reference Tab

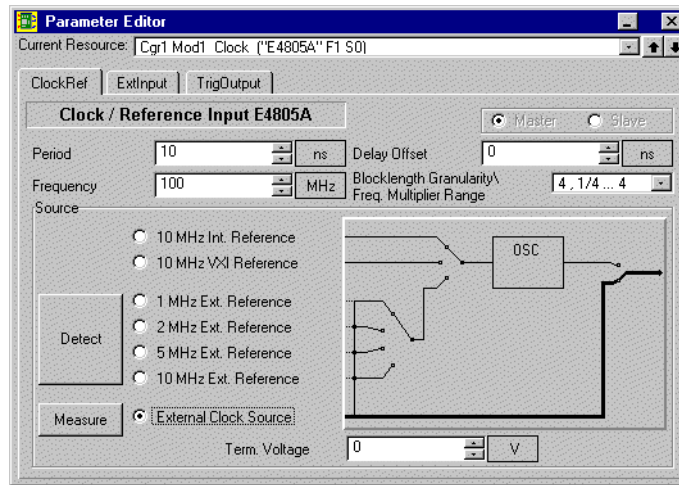
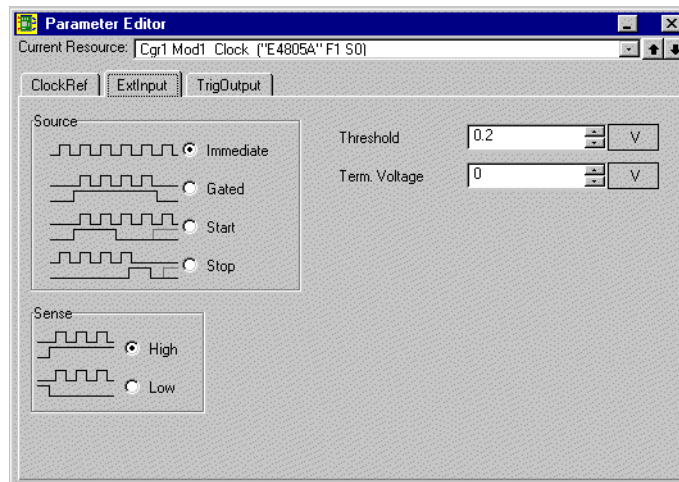


Figure 5 External Input Tab



**IMMEDIATE:**

The timing system is started either manually by the user with a mouse click or by a software command.

**GATE:**

The timing system is armed by pressing the run button. The GUI displays “HALTED” as long as the system is not started. The timing system is started or stopped according to the chosen polarity. The system is stopped, when the sequence terminates. The GUI displays “RUNNING”, “HALTED” or “STOPPED”. See the restrictions for stopping the system, below.

**START:**

The timing system is armed by pressing the run button. The GUI displays “HALTED” as long as the system is not yet started. The timing system is started with the first edge according to the chosen polarity. The system is stopped, when the STOP button is pressed or the sequence terminates.

**STOP:**

The timing system is started by pressing the START button. The timing system is “HALTED” by the first edge according to the chosen polarity and “STOPPED”, when the STOP button is pressed or the sequence terminates. See the restrictions for stopping the system, below.

**Using the External Input:**

The external input is used to start and stop the timing system of the HP81200. The state of this input is sampled once a system period. When the CLK input is used as EXTERNAL CLOCK SOURCE, this is the system period.

When using the EXT INPUT without an EXTERNAL CLOCK SOURCE at the CLOCK/REF INPUT, the signal must be applied for a time greater than the system period.

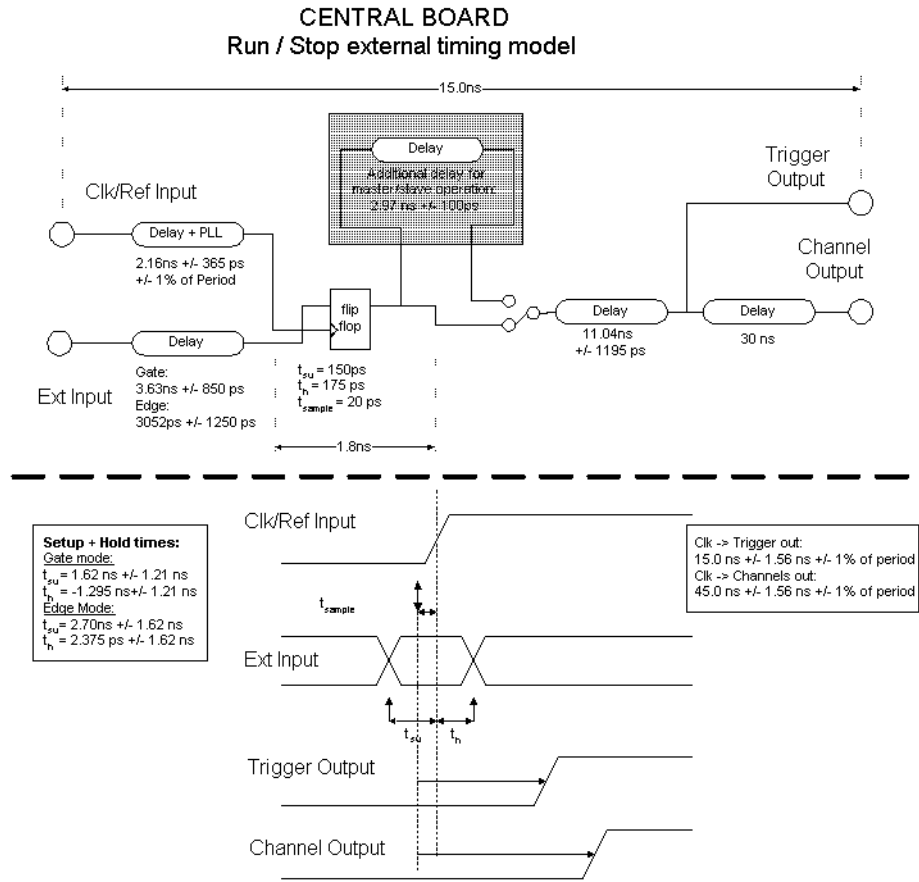
When using the EXT INPUT with an EXTERNAL CLOCK SOURCE at the CLOCK/REF INPUT, setup and hold time must be considered for predictable timing.

**NOTE**

Setup and hold time violations may influence the HP 81200 System only in the aspect that the system will generate consistent relative timings for data generation and data capture, but the absolute timing, related to the external input, may vary of about +/- 1 system period.

Starting the system via the EXT INPUT has no restrictions. All internal pipelines are prefilled so that the first Signal comes out after (+/- 1 system period) + 45 ns + Output delay.

**Figure 6** Delays



When stopping the system via the EXT INPUT some restrictions apply. The timing system is stopped immediately, even if the period and delay of a bit has not completed. The consequence is, that the word at the output might not be aligned during such a stop. After a restart (in GATE MODE) the bits are realigned again.

When stopping the system ensure that the output word is stable for a longer time. Use a PAUSE segment for that purpose. The pause should be (system period \* frequency multiplier \* 32 + maximum delay) long, where the maximum delay is the maximum delay of all involved channels.



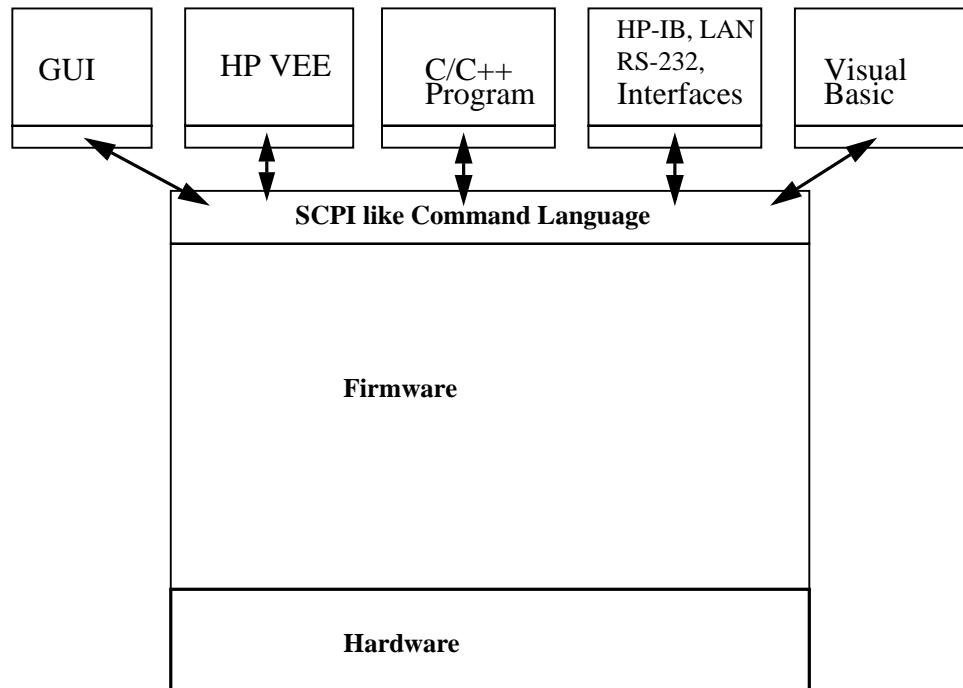
## Software Structure

The software is a client-server architecture. The HP 81200 System can be controlled by one of four different interfaces:

- Graphic User Interface->This control method uses graphical windows to provide the user with a means of setting up a virtual DUT just by pointing and clicking on a mouse button.
- HP VEE interface->This control method uses HP VEE for communicating with the HP 81200 System remotely.
- C / C++ Application Programming Interface (API)->This provides a mechanism for communicating with the HP 81200 System. An application uses the API to send command strings (based on the SCPI command language) to the instrument.
- HP-IB, LAN, RS-232 Interface -> Allows remote control of the HP 81200 System via the HP-IB interface
- Visual Basic Application Programming Interface (API)->This provides a mechanism for communicating with the HP 81200 System. An application uses the API to send command strings (based on the SCPI command language) to the instrument.

All interfaces such as the GUI, HP-IB, HP VEE or C/C++ programs use an ASCII driven serial interface protocol to communicate with the firmware. The ASCII driven serial interface protocol used with the HP 81200 System is based on the SCPI command language.

**Figure 7**                      **Software structure: Client-Server Architecture**



An example of how to use the GUI interface is given in [Chapter 2 “Setting Up Concepts” on page 29](#) of this guide.

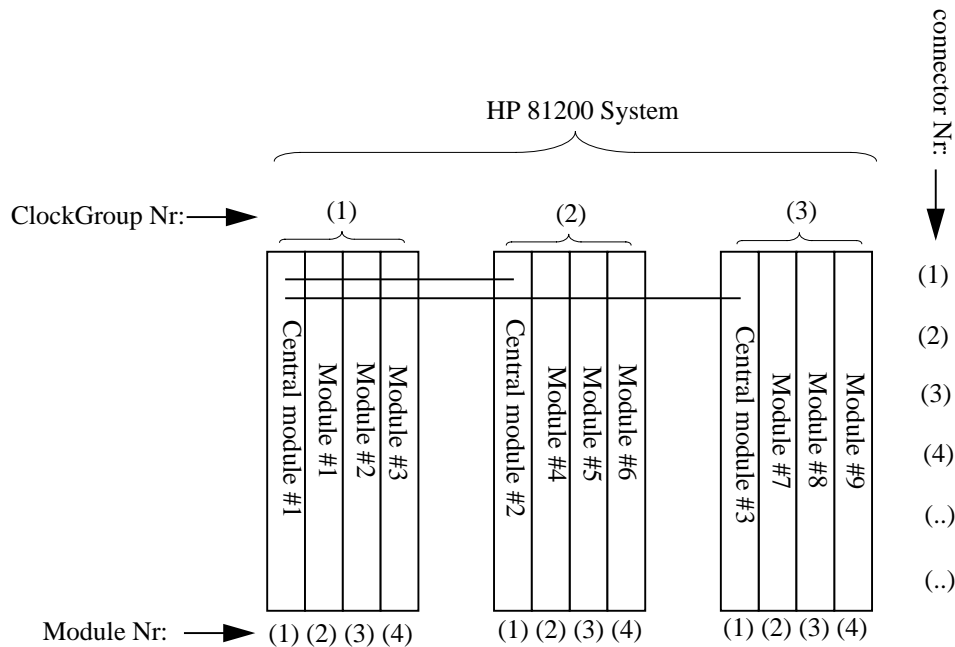
An example of how to use the HP-IB interface is given in [“Controlling the HP 81200 using HP VEE” on page 55](#) of this guide.

## Hardware Resources

The hardware resources (pool) is comparable to a traditional instrument. Here the instrument is seen as a collection of modules which provide several signal connectors. Parameters that can be modified on a connector-level, such as levels, timings, etc, are handled here.

A Digital Stimulus Response (DSR) instrument can consist of multiple clock groups. Each clock group consists of modules which in turn consist of connectors. The following diagram illustrate the numbering system used to address a system, module or connector.

Figure 8



## Connectors

A connector represents an output or input connector on the front panel of a module. A connector may be connected to a terminal of the DUT.

All present connectors are numbered from top to bottom of a module. The NORMAL and the COMPLEMENT output connector of an front-end are counted as one output connector.

## Labeling

In the GUI a connector is represented as a label using the following rules.

The first letter and digit is the “clock group” number, the next letter and digit represent the module and the last letter and digit represent the connector number.

E.g. C1 M2 C3 stands for <clock group 1> <module 2> < and connector 3>

## Channels

In most cases a channel is the same thing as a connector. The cases where this is not true are caused by a special feature the Channel Addition, where up to four generator outputs, installed in one module of the HP 81200 System, can be digitally combined.

### Channel Addition

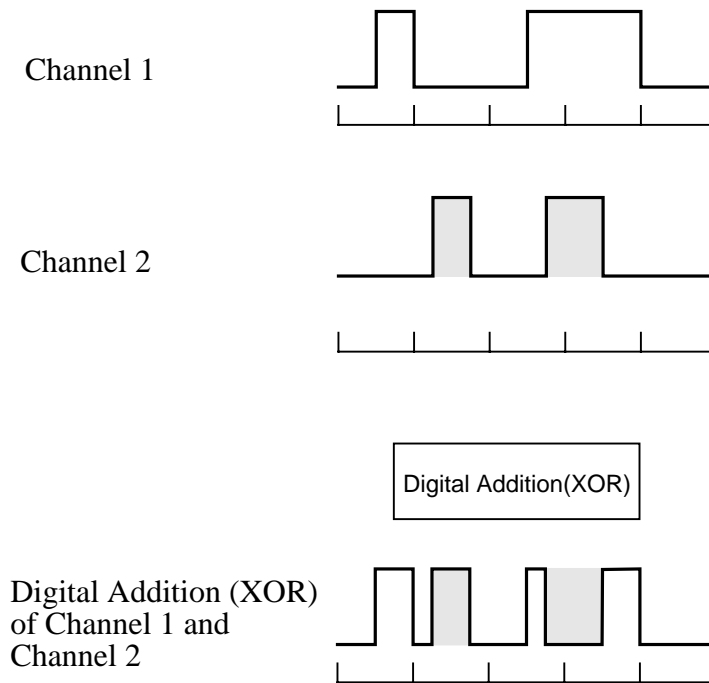
Two or four channels can be digitally added. The digital channel addition is a 'XOR' addition (exclusive OR or modulo 2 addition).

### Digital Addition

Addition takes place before levels are applied to the signals.

Figure 9

Digital Addition



---

## Chapter 2    **Setting Up Concepts**

---

This chapter introduces the setting up concepts of the HP 81200 System with the help of a simple example.

- *“Introduction of the Example” on page 30*
- *“Create Model of Device Under Test (DUT)” on page 33*
- *“Associate Resources of HP 81200 System with DUT” on page 38*
- *“Set System Parameters” on page 40*
- *“Set Signal Parameters” on page 41*
- *“Save Settings” on page 44*
- *“Create the Sequence” on page 46*
- *“Import the Data Segments” on page 49*
- *“RunTest” on page 52*
- *“Comparison of Actual and Theoretical Output Waveforms” on page 53*

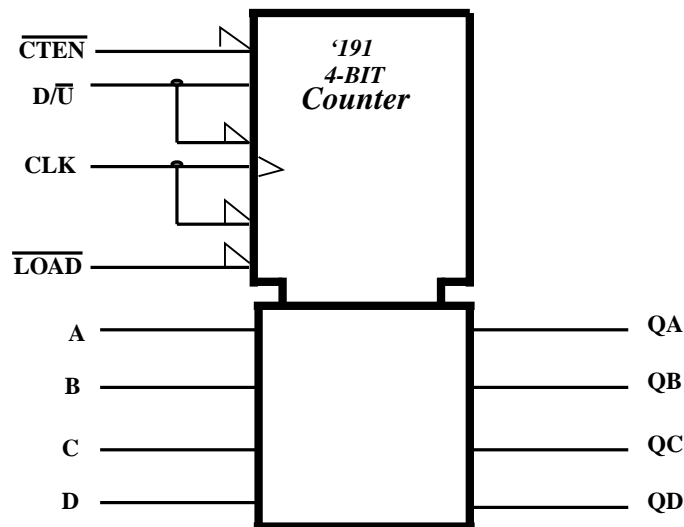
## Introduction of the Example

This example demonstrates the general procedure followed using the HP 81200 System. It will be completed in two steps:

- We will show how to use the resources that are available with the HP 81200 System to create a model of a device under test (*DUT Scheme Modeling and Signal Setup* using the generator capabilities).
- We will then compare the actual output of our model to the theoretical output given for our device under test (*Verification*).

The example will use a type '191 4-bit Counter as the device under test (DUT). The universal logic symbol for the counter is given in Figure 10 below.

**Figure 10**      **Type '191 4-bit Counter**



The outputs of the four flip-flops are triggered on a low-to-high-level transition of the clock input if the enable input,  $\overline{CTEN}$  is low. A high at  $\overline{CTEN}$  inhibits counting. The direction of the count is determined by the level of the down/up,  $D/\overline{U}$  input.

- When  $D/\overline{U}$  is low, the counter counts up.
- When  $D/\overline{U}$  is high, the counter counts down.

A logic table for the counter is shown on [Figure 25 on page 46](#).

### Key Points for DUT Scheme Modeling and Signal Setup

- 1 Create a model of the device under test.
- 2 Associate DUT with HP 81200 System hardware.
- 3 Set system parameters.
- 4 Set signal parameters.

- 5 Save Settings.
- 6 Create sequence segments.
- 7 Import sequence segments.
- 8 Run test.

### **Key Points for Verification**

- 1 Comparison of Actual and Theoretical Output Waveforms.

### **Initial Setup**

- 1 Turn on the HP 81200 System using the power-switch.
- 2 The HP 81200 System should boot and auto-logon as User DVT.

The HP 81200 System software should now start automatically.

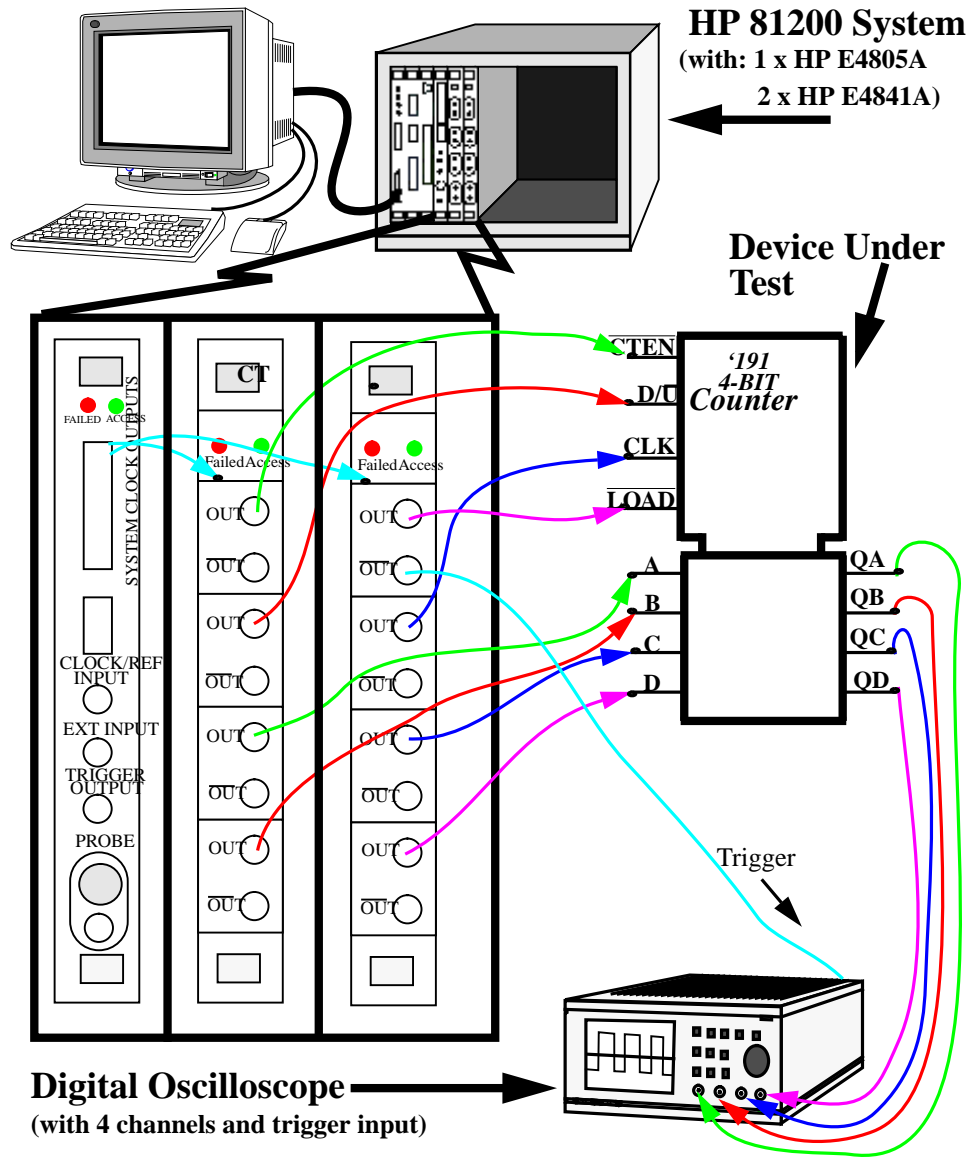
### **Test Setup**

To perform the example we need:

- An HP 81200 System with three modules - two HP E4841A generator modules and one HP E4805A central clock module (see note below).
- Monitor, keyboard and mouse.
- A two/four Channel Oscilloscope (to see the actual signal waveforms).
- Device Under Test (DUT)- type '191 4-bit Counter.
- SMA cables, qty. of 13.

Connect up the system as shown in Figure 11.

Figure 11 Hardware Test Setup for this example





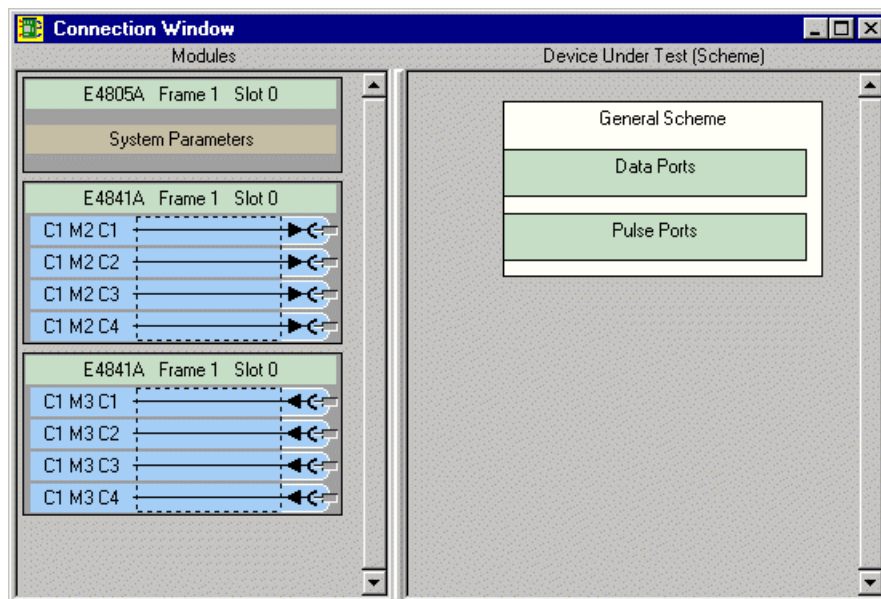
## Create Model of Device Under Test (DUT)

The HP 81200 System is used by creating a model of a DUT in the software and associating the physical resources of the instrument with this ‘virtual’ DUT. The connections between the terminals of the virtual DUT and the virtual instrument correspond exactly to the physical connections between the DUT and the HP 81200 System.

The terminals of a DUT can be grouped into ports, such as data ports and control ports.

A ‘virtual’ DUT is constructed from a scheme (or template). A scheme provides several ports. In the Connection Window of the HP 81200 System, the model of the DUT is created from a General Scheme (See Figure 12).

**Figure 12** Typical Default General Scheme (in Connection Window)



The General Scheme provides four types of ports:

- Data Port. This provides a sequence of segments. Segments can consist of stored data or algorithmic data. Sequences of segments can be repeated and triggered on events.
- Pulse Port. This provides a way to have a pulse generated within the system without the need to setup any data. It is also used for setting up clock signals.

Ports are added within the General Scheme for each group of signals of interest in the DUT, in order to set up the model that will be used by the HP 81200 System.

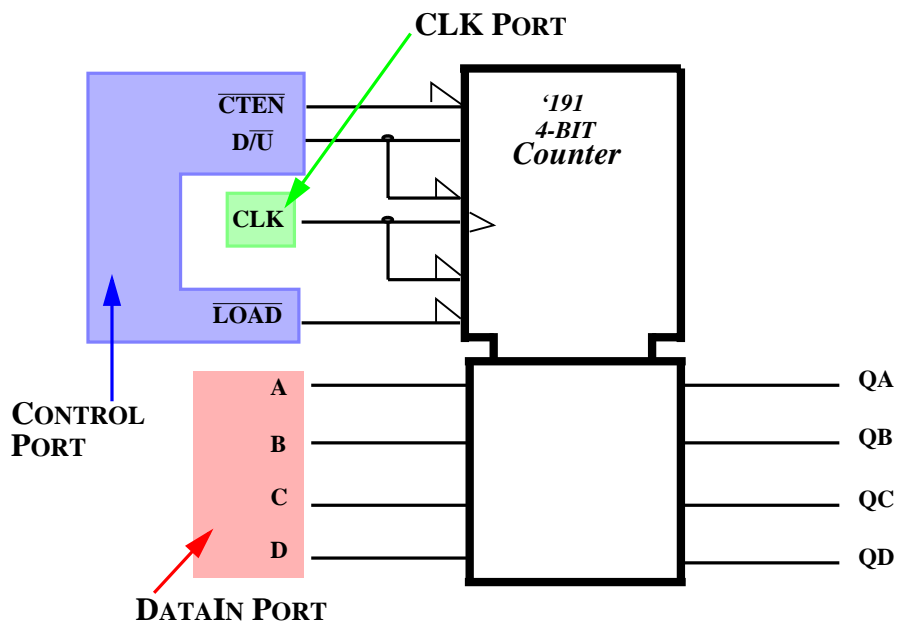
To model the ‘191 4-bit Counter we need the following:

- two data ports:
  - a 3-bit wide Control port....(for  $\overline{CTEN}$ ,  $D/\overline{U}$ ,  $\overline{LOAD}$ )
  - a 4-bit wide DataIn port....(for A, B, C, D)
- a clock port with one terminal....(for Clk)

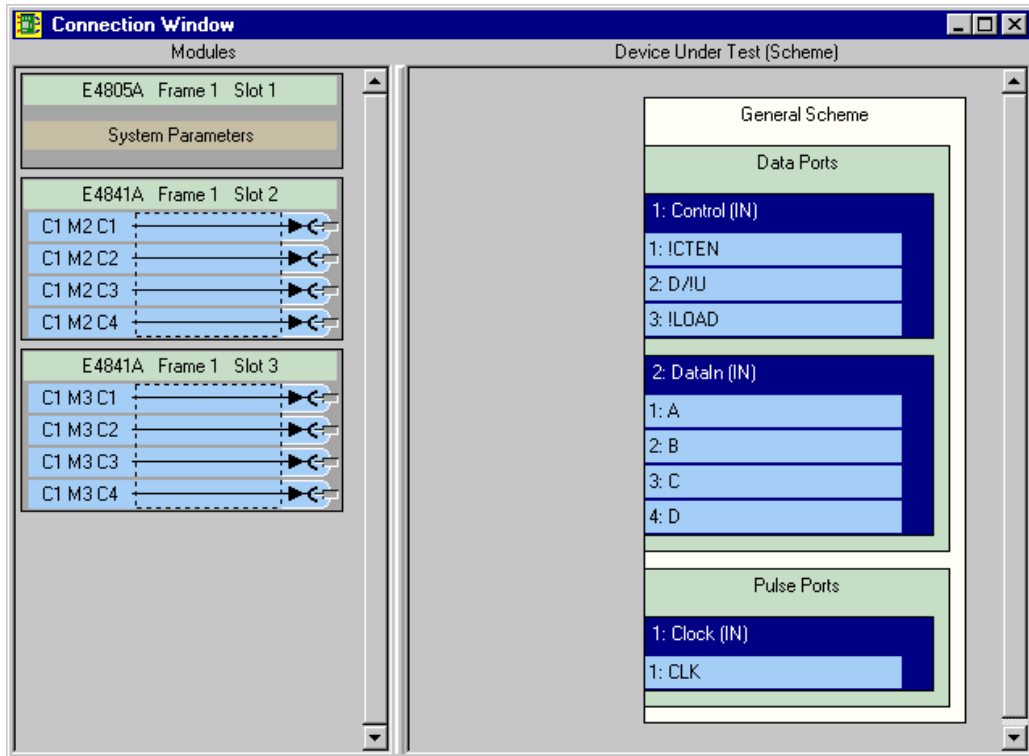
Figure 13 shows how the DUT will be modeled with the HP 81200 System.

Figure 14 shows how the General Scheme will look when all the necessary ports will be created.

**Figure 13** HP 81200 System Model of DUT




**Figure 14** General Scheme for DUT (type '191 4-bit Counter)



**NOTE** Throughout this example, we will use the ! character to indicate the complement output in cases where we cannot draw in the 'bar' over the chosen name. (This can be seen for  $\overline{CTEN}$ ,  $D/\overline{U}$ , and  $\overline{LOAD}$  in the above screenshot: ie.  $!CTEN = \overline{CTEN}$ ).

## Opening the Connection Window

If the Connection Window is not yet active, open it now. This can be done in two ways:

- selecting the **Go->Connection Window** menu item.
- clicking on the Connection Window toolbar button .

## Creating the various ports

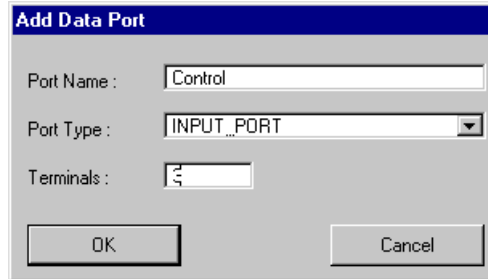
### Adding a data port:

- 1 Right-click on **Data Ports** in the Connection Window.
- 2 Click on **Add Port**.
- 3 Under **Port Name** enter 'Control'.
- 4 Select the **Terminals** option and enter the value 3.

- 5 Click on **OK** to close window.

After completing the above steps, we have now added a 3-bit wide input data port named Control. Repeat the above steps to create another 4-bit wide port called DataIn.

**Figure 15** Add data port window

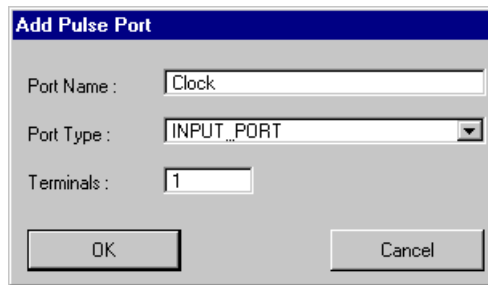


(See also “*Add Port dialog box*” on page 44 of the Reference Guide).

### Adding the Clock signal in a ‘clock’ port:

- 1 Right-click on **Pulse Ports** in the Connection Window.
- 2 Click on **Add Port**.
- 3 Change the port name to **Clock**.
- 4 Select the **Terminals** option and enter the value **1**.
- 5 Click on **OK** to close the window.

**Figure 16** Adding the ‘Clock’ port in the pulse port window



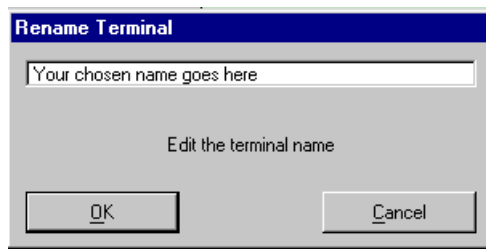
### Renaming the terminals.

To make the connections of the various ports to their corresponding channels more intuitive, it is sometimes useful to rename the terminals that make up these ports. For example, it would be very easy to see at a glance what waveform corresponded to what channel if the terminal name reflected the actual color of the trace on the screen of an oscilloscope.

- 1 Right-click on the terminal that you want to rename.
- 2 Choose **Rename Terminal** from the pop-up menu.
- 3 Type in the name you want to call your terminal, e.g. type !CTEN.

- 4 Click **OK** to confirm the change and close the window.

**Figure 17** Rename terminal window



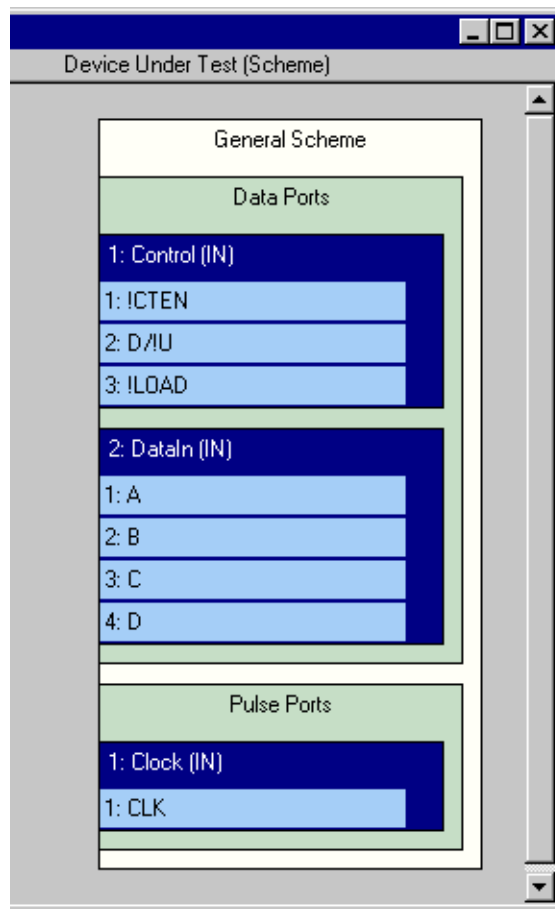
**NOTE**

For the remainder of the example you can choose to use your own terminal names or the names that we have chosen here. The actual naming makes no difference to the operating of the HP 81200 System - it just makes it easier for the user to understand.

(See also *“Renaming a Port/Terminal dialog box”* on page 46 of the Reference Guide).

After adding the above ports, our General Scheme now looks like the following:

**Figure 18** General Scheme





## Associate Resources of HP 81200 System with DUT

When the system is started the software builds a model of the available hardware (modules and front ends). The HP 81200 System is used by creating a model of your DUT in the software and associating the physical resources of the instrument with this ‘virtual’ DUT. The connections between the terminals of the virtual DUT and the virtual instrument correspond exactly to the physical connections between the DUT and the HP 81200 System, so the final physical connections to make can easily be read by this ‘virtual’ setup.


### Connecting terminals to connectors

- 1 Left-click on the terminal you want to connect and keep the mouse button pressed.

The cursor should change to  to indicate that a terminal is being ‘dragged’.

- 2 Drag across to the chosen connector and release when the connect symbol  appears.

Two differences should now be seen in the connection window:

- the number of the connector should appear to the left of the chosen terminal
- a red connected symbol  should appear to the right of the connector

Repeat the above procedure to connect each terminal to a connector, as given below.

Connect  $\overline{CTEN}$  to C1 M2 C1.

Connect  $D/\overline{U}$  to C1 M2 C2.

Connect  $\overline{LOAD}$  to C1 M2 C3.

Connect A to C1 M3 C1.

Connect B to C1 M3 C2.

Connect C to C1 M3 C3.

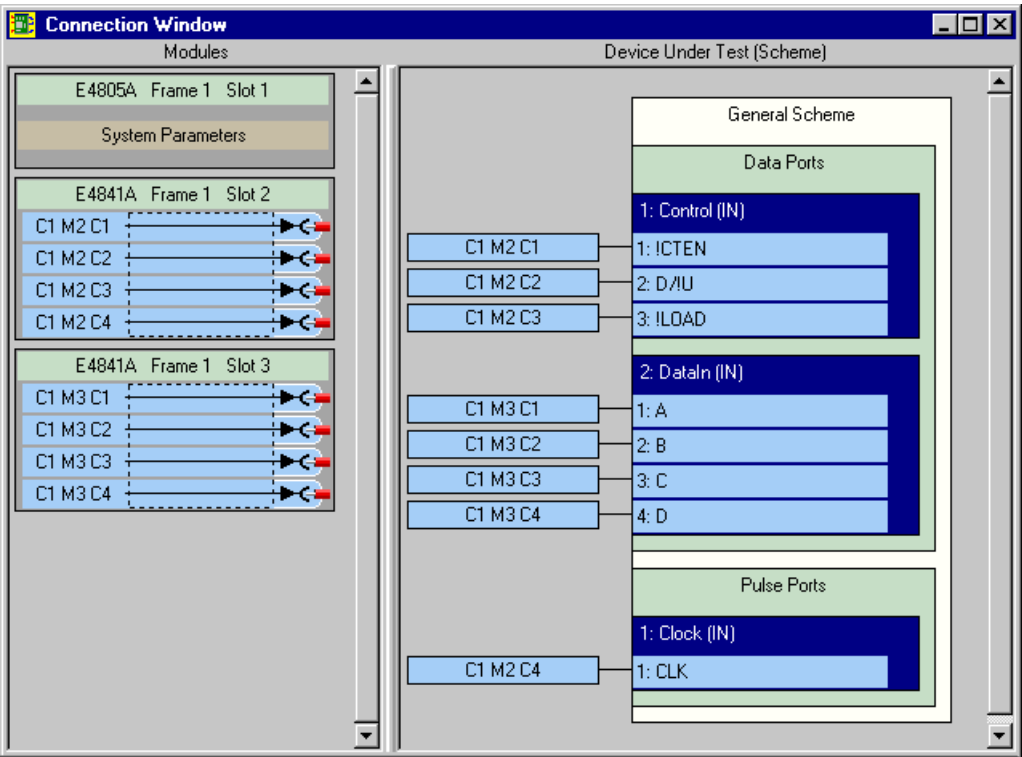
Connect D to C1 M3 C4.

Connect CLK to C1 M2 C4.

Figure 19 shows the connection after all connections have been made.

(See also “*Create a model of the Device Under Test (DUT)*” on page 39 and “*Associate Resources of HP 81200 System with DUT*” on page 38 of the Reference Guide).

Figure 19 Connection window



## Set System Parameters

Now that we have created a model of the DUT, we need to define the way our clock will operate. We do this using the Parameter Editor, which is used to configure the system clock module (ie. the HP E4805A module). We will use a frequency of 20 MHz (equal to a period of 50 ns) to correspond to the given specifications for the '191 4-bit Counter.

Also, as we only have one clock module, we can set this to be our 'master' clock module. (For more in-depth details on the Parameter Editor, see "[Parameter Editor](#)" on page 55 of the Reference Guide).

The Parameter Editor can be accessed in three different ways:


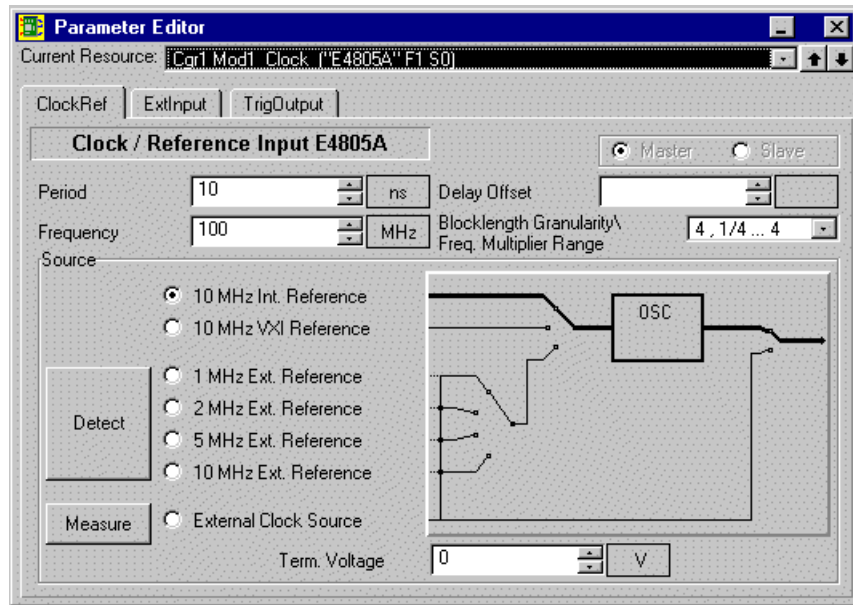

- selecting the **Go->Parameter Editor** menu item.
- double-clicking on **System Parameters**, which is at the top of the Module area in the Connection window.
- clicking on the Parameter toolbar button  does the same operation.

Figure 20 Parameter Editor for Central Clock



### Setting up the clock resources:

- 1 Under **Period** enter the value 50. To change the units, double-click on the box to the right of **Period**, and choose the required units. Alternatively, right-click in the box and choose the appropriate units from the popup menu. For this example we will use nanoseconds -ns. (See also "[Unit and Step Size Adjust](#)" on page 28 of the Reference Guide).
- 2 Click on the **Master** radio button.
- 3 Close the **Parameter Editor** by clicking on  at the top right-hand side corner of the window.



## Set Signal Parameters

Now that we have specified our central clock resource for the DUT, we need to setup the signal parameters for each input of the DUT. These parameters include delay, pulse width/duty cycle and also level parameters such as high level or low level. Signal parameters may be set up for each port or for each terminal. More complex signals may be produced by combining two data sources (channels). This allows signals with glitches, distorted transitions or multiple levels to be produced using channel addition. The complete settings for a virtual DUT may then be saved in the system's database.

The signal parameters are set up using the Parameter Editor. This window can be accessed in three different ways:


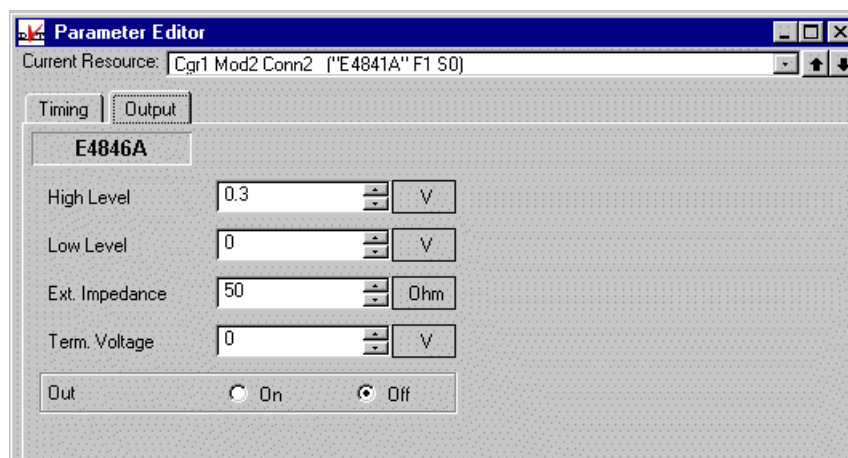
- selecting the **Go->Parameter Editor** menu item.
- double-clicking on a port/connector/terminal in the Connection window.
- clicking on the Parameter Editor toolbar button  .

Figure 21

Parameter Editor



### NOTE

We are going to use the Terminal Editor because we want to edit the signal parameters, as applied to the DUT inputs. Within the HP 81200 System, the term ‘Terminal’ is used to represent signals at the DUT ie. what is at the end of the cables that come from the modules that make up the HP 81200 System (For more details on the differences between a connector, a port and a terminal, see [“Using the HP 81200 System” on page 17](#) of this guide).

- 1 Click on the pull down arrow in the **Current Resource** field.  
This activates a drop-down list, showing all connectors available. Click on the first connector to select it. Using the up and down buttons we can access each connector that is available.
- 2 For each of the connectors that are available, we need to set up the signal parameters for each one, to model our DUT. The table below, Figure 22, gives a summary of the values that we are going to use for each terminal. Also parameters such as levels, termination voltage and external impedance which have values common to all terminals are listed after the table.

**Figure 22** Signal Parameter Values

	<b>Terminal</b>	<b>Format</b>	<b>Delay</b>	<b>Width/ Duty-cycle</b>
Control Port	!CTEN(Control)	NRZ	0	0
	D!/U(Control)	NRZ	0	0
	!LOAD(Control)	NRZ	0	0
DataIn Port	A(DataIn)	NRZ	0	0
	B(DataIn)	NRZ	0	0
	C(DataIn)	NRZ	0	0
	D(DataIn)	NRZ	0	0
Clock Port	CLK(Clock)	RZ	0	50%,Hold Duty Cycle

- All **High-Levels**=2.5V.
- All **Low-Levels**=0V.
- All **External Impedances**=50 Ohms.
- All **Termination Voltages**=0V

**Setting the Signal Parameters for !CTEN(Control)**

- 1 Bring !CTEN(Control) up on the **Current Resource** field, using either the drop-down list or the Up/Down buttons.
- 2 Select the **Timing** tab, then left-click into the **Format** input field and choose NRZ from the popup-menu.
- 3 Select the **Output** tab. Under **High Level**, enter the value 2.5 V. This can be done in two ways:
  - a Highlight the actual value by left-clicking and dragging over the value. The display should now be 'blacked-out', allowing you to type directly into the window, over-writing the highlighted value.
  - b We can increment or decrease the actual value using the Up/Down arrows to the right of the input field. To set the Step Size that the incrementations take, right-click on the units displayed to the right of the input field, and choose **Step Size** from the popup-menu. Click on **0.1** to choose this step size.  
Alternatively, double-clicking on the units display will bring up the Units and Step Size Adjust window, where both the units and step size can be chosen by clicking on the appropriate radio button.

(See also “*Unit and Step Size Adjust*” on page 28 of the Reference Guide).

- 4 Change the **Low Level** setting to 0 V.
- 5 Change the **Ext. Impedance** to 50 Ohms.
- 6 Change the **Term. Volt.** to 0 V.
- 7 In the Data page, click on the **User Defined** radio button.
- 8 Click the ‘On’ radio button for the **Out** output.

This completes the procedure for setting the signal parameters for the !CTEN(Control) terminal. As the D/!U(Control), !LOAD(Control), A(DataIn), B(DataIn), C(DataIn), D(DataIn) terminals use the identical values, the above procedure can be repeated to set their parameters also.


#### Setting the Signal Parameters for CLK(Clock)

- 1 Bring CLK(Clock) up on the **Current Resource** field, using either the popup-menu or the Up/Down buttons.
- 2 In the **Timing** tab, click on the **Duty Cycle** radio button.
- 3 Enter 50% in the **Duty Cycle** field.
- 4 Change the **Format** to RZ.
- 5 Select the **Output** tab. Under **High Level**, enter the value 2.5 V.
- 6 Change the **Low Level** setting to 0 V.
- 7 Change the **Ext. Impedance** to 50 Ohms.
- 8 In the Data page, click on the **Ones** radio button.
- 9 Click the ‘On’ radio button for the **Out** output.

#### Setting up a Trigger for the Oscilloscope

For our oscilloscope to function correctly, we need to take an output that can be used as a trigger. We will use the complement output of !LOAD as our trigger, as this only changes once within the sequence of the ‘191 4-bit Counter.

- 1 Bring !LOAD(Control) up on the **Current Resource** field.
- 2 In the **Output** tab, click the ‘On’ radio button for the  $\overline{\text{Out}}$  output.

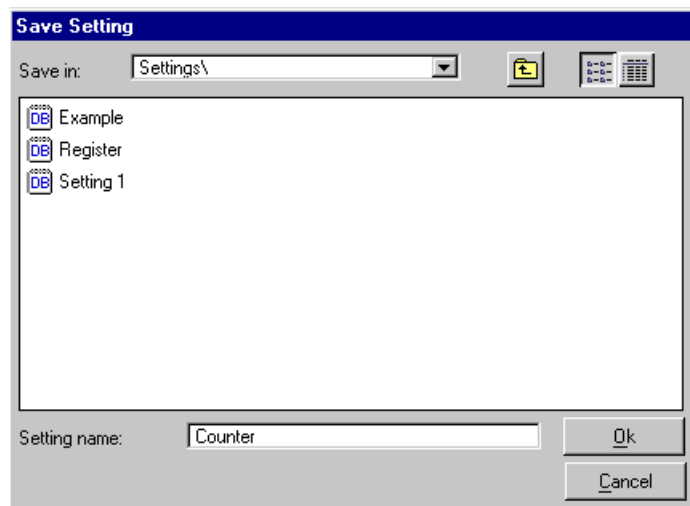
This completes all the settings that we need for our example. We can now exit the Parameter Editor by clicking on  at the top right-hand side corner of the window, to proceed onto the next part of the example.

## Save Settings

After setting up the various parameters in the Parameter Editor, it is now possible to save them, to be referenced in the future.

- 1 From the File menu-item, click on **Save Setting As**.  
This will bring up the Save Setting dialog box shown below. This dialog box displays any previously saved settings that are already in the database.

**Figure 23** Save Setting dialog box.



- 2 For the **Setting Name**, enter 'Counter'.
- 3 Click **Ok** to confirm the choice.

### NOTE

At the moment it is only possible to save settings to the **Settings Pool** within the HP 81200 System database. It will be possible, at a later date, to save settings outside of the database ie. to export them as text files. Also, it is not possible to remove a setting from the database once it has been saved. This feature will be implemented in the near future.

### Opening a Saved Setting

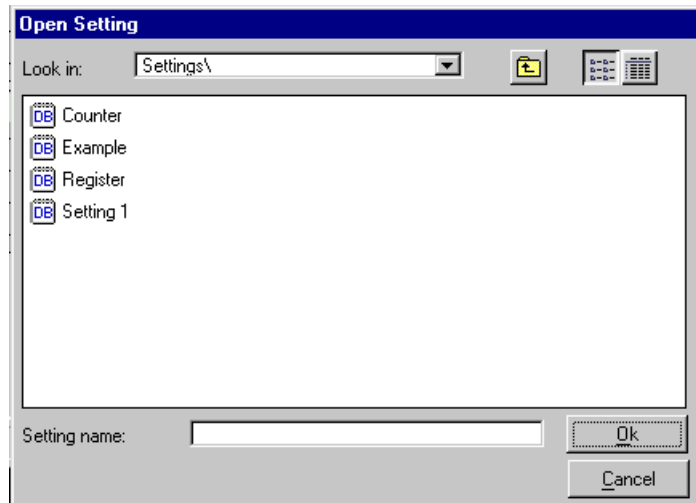
- 1 From the File menu, click on **Open**. Or, it is also possible to open a setting by clicking on the Open

Button  in the toolbar.

This will bring up the Open Setting dialog box shown below.

Figure 24

### Open Setting Dialog Box



- 2 The dialog box will show a listing of all available settings that are stored in the database. Click on 'Counter' to re-open our previous settings.

- 3 Click **Ok** to close the dialog box.

The title of the Main Window should now reflect the name of the setting we are currently using.

## Create the Sequence

When the signal parameters have been set up it's time to set up patterns. The type of pattern that may be applied to a port depends on the type of port. Data ports can output sequences of segments. These sequences can contain loops and may be triggered on events. Segments may be stored in a global segment pool or in the segment pool of the current setting.

For the '191 4-bit Counter, the sequence we want to set up is given in the table below Figure 25. The table shows how we will split the required data into two segments, Cntrl and DataIn.

- the 'Cntrl' segment to handle the !CTEN, D!/U, !LOAD inputs.
- the 'DataIn' segment to handle the A, B, C, D parallel data inputs.

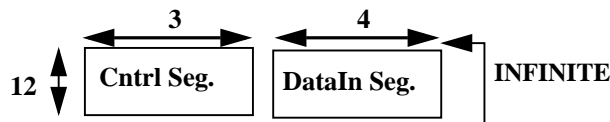
**NOTE** The Clk input does not require any data to be setup.

**Figure 25** Sequence Required for '191 4-bit Counter

Clock cycle	!LOAD	D!/U	!CTEN	Data Inputs			
				D	C	B	A
1	0	1	0	1	1	0	1
2	1	1	0	0	0	0	0
3	1	1	0	0	0	0	0
4	1	1	0	0	0	0	0
5	1	1	0	0	0	0	0
6	1	1	0	0	0	0	0
7	1	0	1	0	0	0	0
8	1	0	1	0	0	0	0
9	1	0	0	0	0	0	0
10	1	0	0	0	0	0	0
11	1	0	0	0	0	0	0
12	1	0	0	0	0	0	0

In this example we will create a sequence which repeats the two segments as shown in Figure 26. We will import the two segments from a text file as the Segment Editor is not yet implemented.

**Figure 26** Sequence Scheme



### Creating the Vector Format Text File

The sequence that we will import in the next section of the example is created as a text file using a normal text editor. This text file has to follow a certain format so that the HP 81200 System is able to process it properly (See “*Vector Import and Export Tool*” on page 105 of the Reference Guide for a more detailed description on the Vector format).

#### Vector Format Text File:

Figure 27

Vector Format Text File

```

:vectorVariablesDefinitions:
{
  :paraPatternVar:
  {
    :name: Cntrl
    :statePar: { {A "01"} }
    :stateSet: A
    :vectorWidth: 3
    :vectors:
    {
      010
      110
      110
      110
      110
      110
      101
      101
      100
      100
      100
      100
    }
    :parameters:
    {
      { _Type (MEMORY) }
    }
  }
}
:vectorVariablesDefinitions:
{
  :paraPatternVar:
  {
    :name: Cntrl
    :statePar: { {A "01"} }
    :stateSet: A
    :vectorWidth: 4
    :vectors:
    {
      1101
      0000
      0000
      0000
      0000
      0000
      0000
      0000
      0000
      0000
      0000
      0000
    }
    :parameters:
    {
      { _Type (MEMORY) }
    }
  }
}

```

Pattern for the Control  
Signals !CTEN, D!/U, !LOAD

Pattern for the Data Signals  
A, B, C, D

Figure 27 above shows the format for creating the Vector text file.

Required in the file:

1. an overall header (**:vectorVariablesDefinitions: {...}**).
2. an initial segment tag (**:paraPatternVar: {...}**).
3. a name for the segment (e.g. **:name: Cntrl**).
4. the state coding for the segment (e.g. **:statePar: { {A "01"} }**)
5. the vector width [represents the number of signals grouped] (e.g. **:vectorWidth: 3**)
6. a line marking the beginning of the vector listing (**:vectors: {...}**).
7. the actual vectors (data).
8. the segment type definition (e.g. **:parameters: { { \_Type (MEMORY) }}**).

**NOTE**

Items 2-5 above are required for each new segment that is created.

The vectors that have been entered into the example shown in Figure 27 correspond to the actual data required to create the sequence depicted in *Figure 25 on page 46*.

- 1 Using a normal text editor (e.g. Notepad), create the example shown in *Figure 27 on page 47*.
- 2 Save the text file as 'sample.txt' to C:/HP81200/DSR/Samples/Segments/.
- 3 The above example (Figure 27) is already saved as a text file to the HP81200 program group. It is saved as 'sample.txt'. To view the file and make any necessary changes, locate it in:  
C:/HP81200/DSR/Samples/Segments/sample.txt

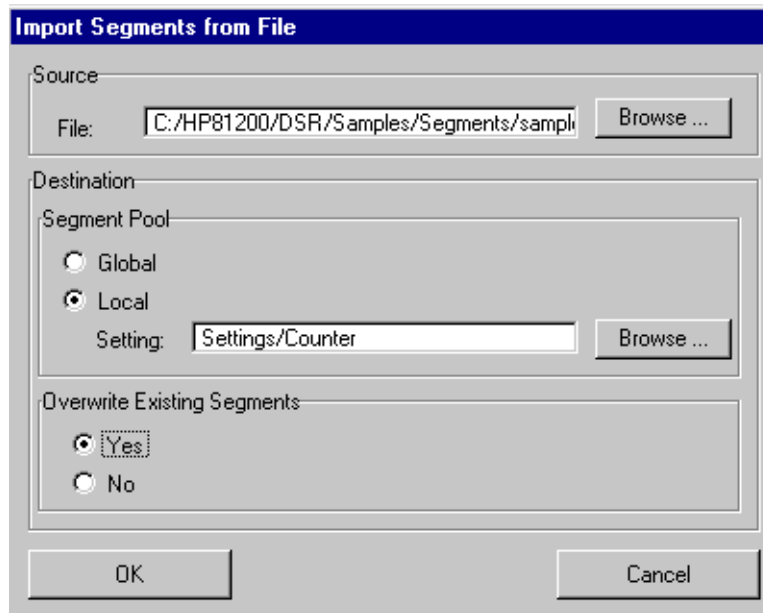


## Import the Data Segments

After the sequence has been created as a text-file, it must be imported into the database of the HP 81200 System. This is done using the following steps:

- 1 Click on **Import Segment** in the **File** menu item.  
 This will bring up the Import Segments dialog box.

**Figure 28**            **Import Segments dialog box**




- 2 Use the browser to locate the required segments at C:/HP81200/DSR/Samples/Segments/sample.txt.
- 3 For the destination, click on the **Local** radio button in the Segment Pool page.
- 4 Use the browser to select **Counter** from the Settings Pool.
- 5 Click the **Yes** radio button in the Overwrite Existing Segments page.  
 The database has now been updated with the new segment definitions.

### Updating the Sequence Editor

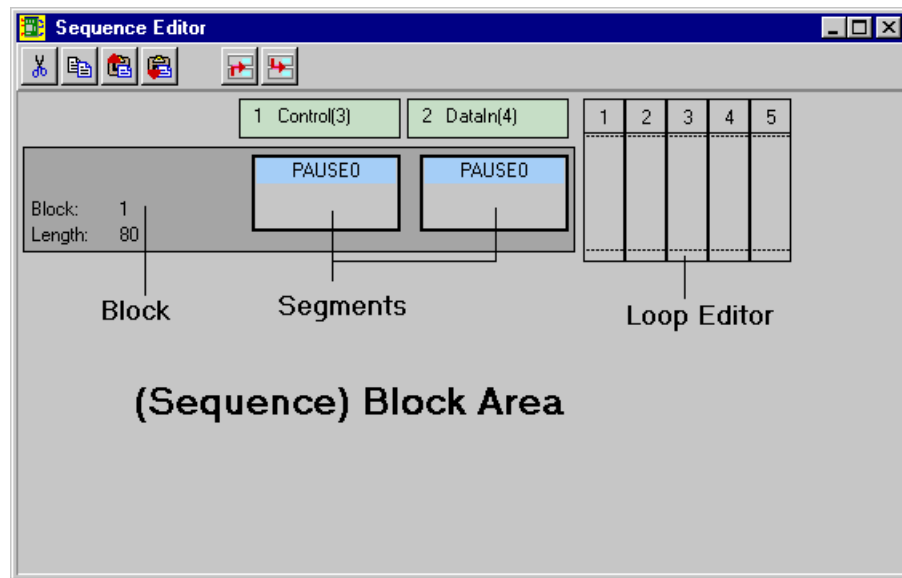
After importing the segment definitions, we must now update the Sequence Editor to display the segments we want to use.

- 1 Select the **Go->Sequence Editor** menu item. Or click on the Sequence Editor toolbar

button  to do the same operation.

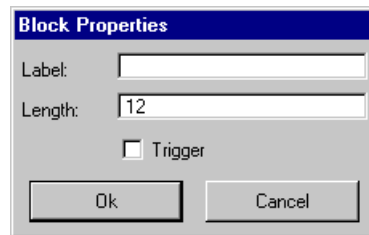
We are now presented with the Sequence Editor window, as shown in Figure 29.

**Figure 29**                    **Sequence Editor**




- 2 We must first set the Block Length. Right-click on the block.
- 3 From the popup-menu click on **Properties**.
- 4 Enter the value **12** for the block **length** in the **Block Properties** dialog-box.

**Figure 30**                    **Block Properties dialog-box**



- 5 Click **OK** to close the dialog-box.
- 6 Right-click on the left-hand side segment diagram.  
-a popup-menu should now appear
- 7 From the popup-menu click on **Select Segment**.
- 8 In the browser choose **Ctrl**.
- 9 Click **OK** to close the dialog-box.
- 10 Click on the right-hand side segment diagram.
- 11 Repeat step 7 above and in the browser choose **DataIn**.
- 12 Click **OK** to close the dialog-box.  
After following the above steps, the names of the two chosen segments should appear in the Sequence Editor and a new icon should appear on the toolbar.

This is the Download Sequence  toolbar button.

13 Click on the Download Segment toolbar button.

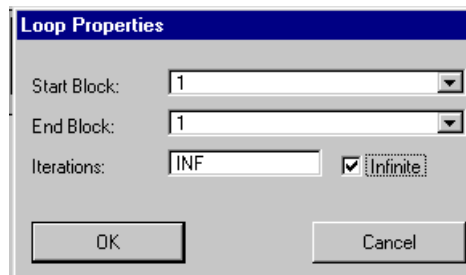
The Sequence Editor should now be updated with the new segments.

### Loop Properties

Now that we have selected our sequence from the HP 81200 System database, we must instruct the system how many times we want this sequence iterated. We do this using the **Loop Editor**.

- 1 Right-click on the Loop Editor located to the right of the segment blocks.
- 2 From the popup-menu, click **New Loop**.  
This will bring up the **Loop Properties** dialog-box.

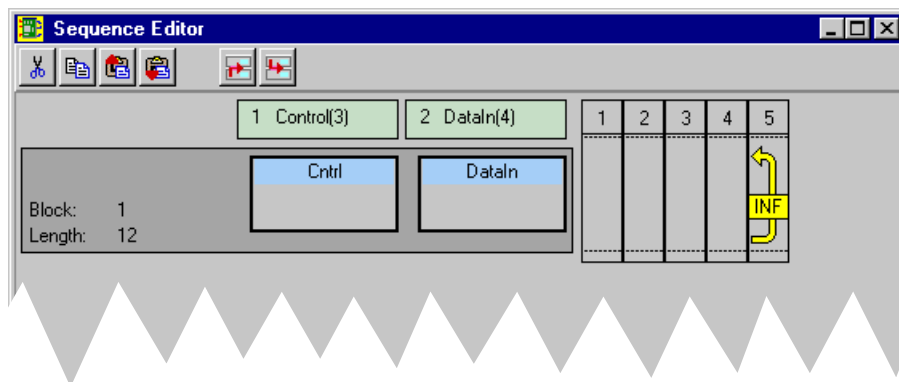
Figure 31 Loop Properties dialog-box



- 3 In the **Start Block** field. enter 1.
- 4 In the **End Block** field. enter 1.
- 5 In the **Iterations** field. click on the **Infinite** check-box.
- 6 Click **OK** to confirm the selection.

We have now instructed the HP 81200 System to repeat our sequence infinitely. The completed sequence is shown in Figure 32.

Figure 32 Completed Sequence



## RunTest

After creating a model of our DUT and setting all the required parameters, we are now ready to run our test. To do this, go to the Main window and click on the Run button. The status area should reflect this input by changing from ‘Stopped’ to ‘Running’.

Figure 33 Main Window Icon Bar

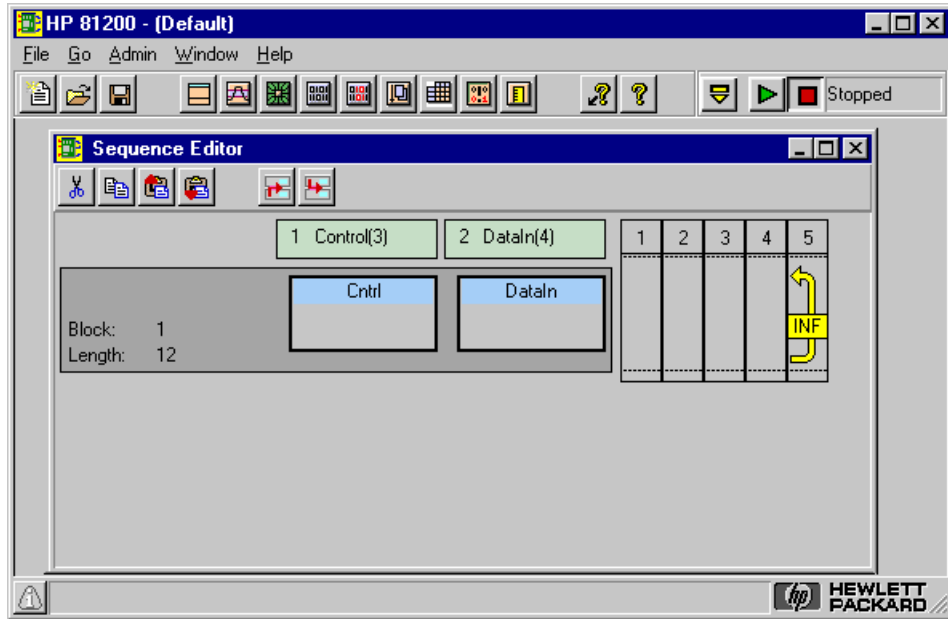


Figure 34 Test Stopped



Figure 35 Test Running



## Comparison of Actual and Theoretical Output Waveforms

To verify that the '191 4-bit counter is operating correctly, we will compare the theoretical output waveforms given in the specifications with the actual output waveforms from the DUT, as shown on a Digital Oscilloscope.

Shown in Figure 36 below are the theoretical output waveforms for the '191 4-bit Counter.

Shown in Figure 37 below are the actual observed output waveforms for the '191 4-bit Counter.

Figure 36 Theoretical Output Waveforms (from given specifications)

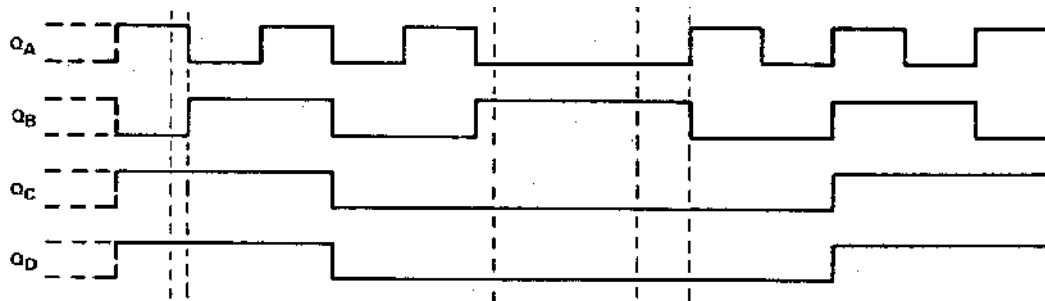
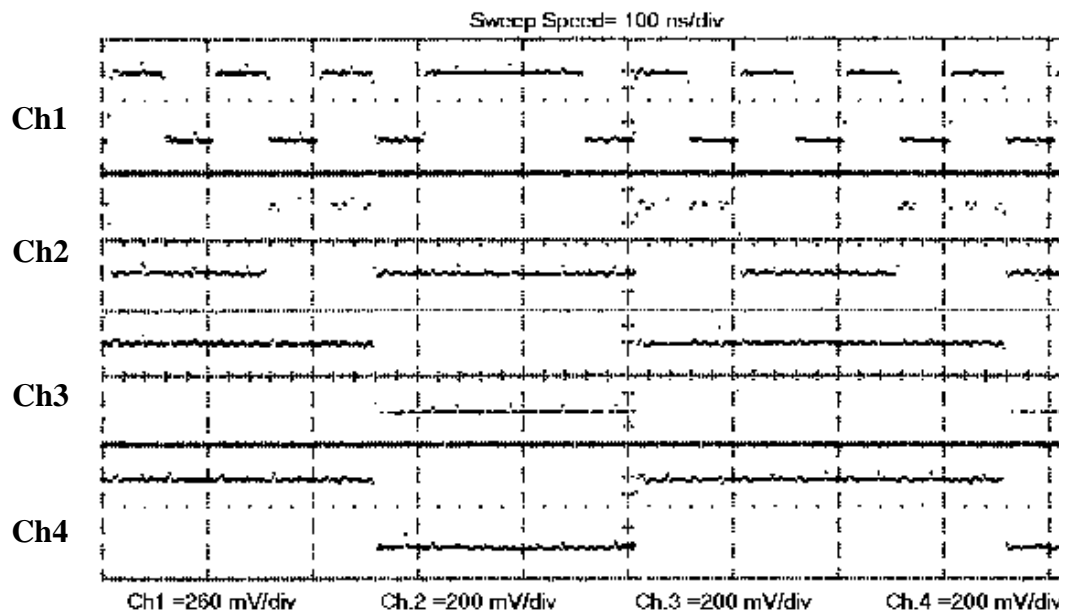


Figure 37 Actual Observed Output Waveforms (from a Digital Oscilloscope)



Ch1 corresponds to the QA output. Ch2 corresponds to the QB output. Ch3 corresponds to the QC output. Ch4 corresponds to the QD output.

As can be seen, these two Figures correlate, verifying that the '191 4-bit Counter is functioning properly under nominal conditions.



---

## Chapter 3    **Controlling the HP 81200 using HP VEE**

---

This chapter

The following are described:

- *“A simple Example using VEE” on page 56*

## A simple Example using VEE

This example gives an introduction of how to control the HP 81200 System using HP VEE via HP-IB.

### General Information

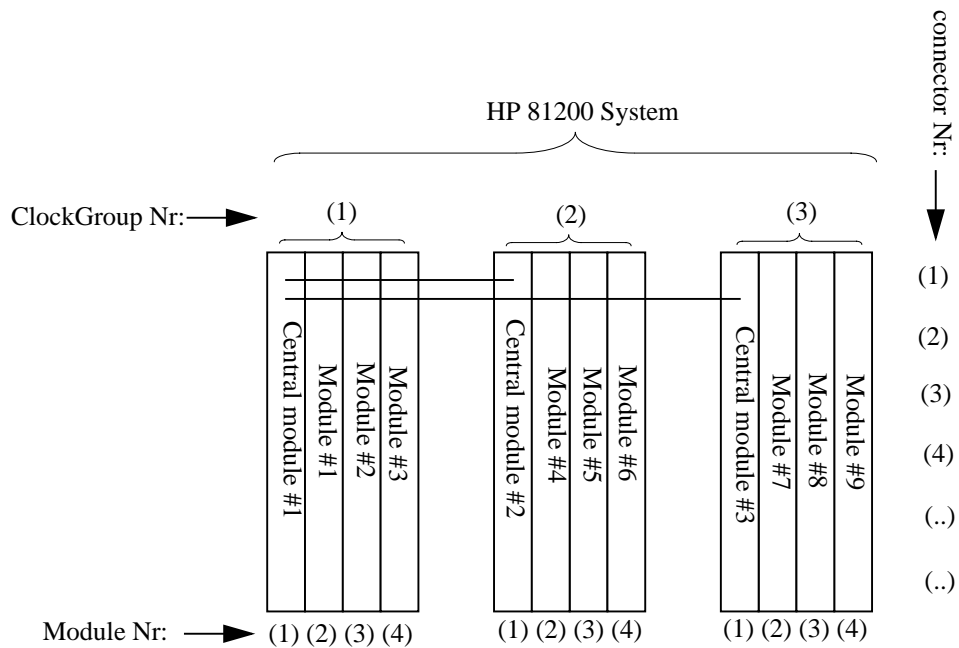
The HP 81200 System's hardware resources (clock modules, and generator/analyzer modules) can be configured that there is one or are more instruments present, to talk to via GUI, or HP-IB.

### Hardware Resources

The hardware resources (pool) is comparable to a traditional instrument. Here the instrument is seen as a collection of modules which provide several signal connectors. Parameters that can be modified on a connector-level, such as levels, timings, etc, are handled here.

A Digital Stimulus Response (DSR) instrument can consist of multiple clock groups. Each clock group consists of modules which in turn consist of connectors. The following diagram illustrate the numbering system used to address a system, module or connector.

Figure 38





## Steps to do for Remote Programming the HP 81200 System by HP VEE via HP-IB

- 1 Checking for HP-IB connection.
- 2 Checking for virtual instrument names.
- 3 Creating a <Handle> for a virtual instrument.
- 4 Sending commands to the virtual instrument.
- 5 Finally, destroying the <Handle>.

### Test Setup

The HP 81200 System connected via HP-IB to a PC running HP VEE (in this example HP VEE Version 4.0 is used).

**NOTE** When using the HP SICL then as default the HP-IB controller address is 21. Please do not use this address for the HP 81200 System (default address is 11), or any other instrument you connect to the HP-IB bus.

### Sample Program

There is a sample program available. The name of the sample program is “exdvtinh.vee”. There is a second file “exdvtihc.vee”, which is the same program, but has comments added to each object in the program flow. If you have chosen the default installation path, the sample programs are available in the directory c:\hp81200\dsr\samples\hp\_vee

**NOTE** The programs are designed under HP VEE Version 4.0

**Important!** If you set up a new HP VEE program for the HP 81200 System perform the following hints and recommendations:

Make sure that the HP 81200 System is not set to HP-IB controller.

Make sure that when you set up a new I/O Device with the instrument manager of HP VEE you have to set END (EOI) on EOL to YES in the Direct I/O tab.

### Procedure

- 1 Connect the HP 81200 System via HP-IB cable to the PC.
- 2 Start HP VEE on the PC.
- 3 Open the program “exdvtinh.vee”, which is available in the directory c:\81200\dsr\samples\hp\_vee (if you didn't changed the default installation path).
- 4 Click on the [Start] object in the program, or on the Run icon in the icon bar.
- 5 A dialog box pops up. In the text entry field there is text highlighted. Overwrite this highlighted text with a handle name as your suggestion. Recommended to start with an underscore, use only alpha characters. The handle should be less equal 12 characters.

**NOTE** You can have a fix handle by deleting the text entry and replacing the variable A by the fix handle name.

- 6 In this program a new setting will be created, this is recommended to start from a known condition and to avoid too many parameter conflicts.
- 7 Each new command can be added in one I/O box for HP 81200, titled HP 81200 (@7xx) { where xx is the HP-IB address of the HP 81200 System. See Note under **Test Setup**.
- 8 It is recommended to send period/frequency and block length granularity (MUX factor) in one command line, as well as high and low levels, as these could easily lead to parameter conflicts between the current and the new settings. For Block Length Granularity, Frequency Multiply Range, Memory Depth and max Frequency see the [Table 1 on page 11](#), in chapter 1, Product and Concepts Overview.

For information of the command syntax and the command reference list, please refer to the chapter 2, Command Syntax Description and chapter 3, Command Reference List in the HP 81200 System Reference Guide.

The Reference Guide is available in PDF format and can be displayed with the Acrobat Reader which is installed on the system. You can access the Reference Guide by either clicking on the HP 81200 Docu icon on the desktop, or by clicking on Start and selecting the HP 81200 Documentation from the HP Digital Verification Tools program group.

- A**
    - Analyzer Channels, 12
  - C**
    - C / C++ API, 25
    - Cable Delay Compensation, 12
    - Channel Add, 11
    - Channel Addition, 28
    - Channels, 28
    - Characterize Digital Devices, 10
    - Client-Server Architecture, 25
    - Clock Source Module, 14
    - Concepts Example
      - Example
        - showing setting up, 29
    - Configurations, 16
    - Connector
      - input, 17
      - output, 17
    - Connectors, 27
  - D**
    - Data Generator/Analyzer Module, 14
    - data port, 17, 33
    - Data Streams
      - create, 11
    - delay, 17
    - Digital Addition, 28
  - E**
    - Emulate Real Signals, 10
    - END (EOI)
      - setting for HP VEE, 57
    - EOL
      - setting for HP VEE, 57
    - Error Analysis, 13
    - events, 46
  - F**
    - Front-ends, 15
    - Functional Tests, 12
  - G**
    - Gated, 17
    - general scheme, 17, 33
    - Generator Channels, 12
    - Global segment, 17
    - Graphic User Interface, 25
  - H**
    - Hardware Resources, 12, 27
    - HP VEE, 25
    - HP VEE Program Example, 55
    - HP-IB Interface, 25
  - I**
    - Input
      - connector, 17
  - L**
    - Labeling, 27
    - levels, 17
    - Local segment, 17
  - M**
    - Margin Testing, 13
    - Modules, 14
  - N**
    - Naming Conventions, 18
    - Negative Delay, 12
  - O**
    - Output
      - connector, 17
  - P**
    - Pattern Generation, 10
    - patterns, 17, 46
    - Port, 17
    - port
      - data, 17, 33
      - pulse, 17, 33
    - ports, 17, 33
    - Propagation Delay Compensation, 12
    - pulse port, 17, 33
    - pulse width, 17
  - R**
    - Remote Controlled Operation, 12
  - S**
    - scheme, 17, 33
      - general, 17, 33
    - Segment, 18
    - Segments, 17
    - segments, 46
    - Sequence, 18
    - sequences, 17, 46
    - Setup and Hold Time measurement, 12
    - Software Structure, 25
    - Started
      - external, 17
    - Stopped
      - external, 17
  - T**
    - Terminal, 17
    - terminals, 17, 33
    - Testing at speed, 10
    - Testing up to 1.3 Gbit/s, 11
    - Testing up to 660 Mbit/s, 10
    - timing, 17
  - V**
    - Verify Digital Devices, 10
    - virtual DUT, 17, 33, 38
  - Z**
    - Zero Adjust, 12
-

