PROGRAMMABLE CLOCK APPLICATION NOTE

APPLICATION NOTE AN-234

TABLE OF CONTENTS

INTRODUCTION

IDT5T989x and IDT5T982x devices are the timing devices that can be configured in-system. The devices include EEPROM non-volatile memory to store the user's configuration.

DEVICE OVERVIEW

There are four members in the family. IDT5T9820 and IDT5T9821 are zero delay buffer (ZDB) versions, and are targeted for applications that have no specific propagation delay requirements between inputs and outputs. IDT5T9890 and IDT5T9891 are programmable skew versions, and are targeted for applications that require individual skew adjustment for each output pair.

FEATURES DESCRIPTION

The devices have redundant clock inputs, and each input can support either single-ended or differential types. The device also supports input clock frequencies from 4.17MHz up to 250MHz, and output clock frequencies from 12.5MHz up to 250MHz.

On-chip non-volatile EEPROM allows the user to save the device configuration and load them automatically upon power-up, without having to reprogram the part every time. Both I²C and JTAG interfaces can be used to program the device's volatile registers and EEPROM memory. To be backward compatible with older voltage technology, all control pins and clock inputs are 3.3V tolerant, except for the 3-level (ADDRx) and I²C/JTAG pins. To support Design-for-Testability (DFT), the devices also provide JTAG boundary scanning capabilities.

Each output bank supports four I/O standards: 2.5V LVTTL, HSTL, eHSTL, and 1.8V LVTTL. Each bank I/O standard selection is independent of the other bank. For example, while running one bank at 2.5V LVTTL, another bank can be at HSTL, and so on. The dedicated V_{DDQN} pins to each output bank must be connected to the appropriate voltage level.

The devices generate less than 100ps of cycle-to-cycle jitter, which is an order of magnitude better than other programmable devices in the market. These devices are also compatible with a Spread Spectrum Input Clock, meaning it will track the input's Spread Spectrum Clock (SSC). Each device can bypass the PLL to function as a regular clock fanout buffer.

There is a feedback divider for frequency multiplication of 1-6, 8, 10, 12 and a post-divider of 2 and 4.

Device	Zero Propagation Delay	Complementary Outputs	Output-to-Output Skew Adjustment	Programmable I/O	In-System Programming
IDT5T9820	X			X	X
IDT5T9821	X	X		X	X
IDT5T9890	X		X	X	X
IDT5T9891	X	X	X	X	X

DETAIL FEATURE DESCRIPTION

OUTPUT ENABLE ($\overline{OE}$) FEATURE

For the IDT5T9890 and IDT5T9820, there is a pair of single-ended outputs on each of the first five banks, excluding the differential feedback bank. For the IDT5T9891 and IDT5T9821, there is one differential output for each bank, including the differential feedback bank. Each bank has its own output enable/disable external pin ($\overline{nSOE}$) as well as a corresponding function in Bit 56-52 for serial programming. Both the external pin and corresponding Bit are used in conjunction and function as an NOR where both have to be LOW to enable the output and if either one is HIGH then the output will be disabled.

When the output is disabled, the state of the output can be determined by the external pin OMODE or its corresponding Bit 59. These two are also used in conjunction and function as an NOR where both have to be LOW to tri-state the output and when either one is HIGH then the disabled output would gate to a LOW or HIGH state.

The gated output state will depend on Bit 58. Bit 58 actually determines if the PLL will synchronize to either the positive or negative edge of the reference clock. This ties to the gated output state in such that when the PLL is synchronized to the positive edge, the output will be gated to a LOW state and if synchronized to the negative edge, to a HIGH state. For differential outputs, with positive sync, the true outputs will be gated to a LOW state and complementary to a HIGH state. With negative sync, the true outputs will be gated to a HIGH state and complementary to a LOW state.

The tables below capture the different options and states for enabling/disabling the outputs.

JTAG/ I²C SERIAL CONFIGURATIONS(1)

PD	$\overline{nsOE}$	OMODE	Output States
H	1 or H	1 or H	Gated
H	1 or H	0 and L	Tri-Stated
H	0 and L	X	Normal
L	X	1 or H	Gated
L	X	0 and L	Tri-stated

NOTE:

1. OMODE and its corresponding Bit 59 selects whether the outputs are gated LOW/HIGH or tri-stated. When OMODE is HIGH or the corresponding Bit 59 is 1, the outputs' disable state will be gated. Bit 58 determined the level at which the outputs stop. When Bit 58 is 0/1, the single-ended and true (differential) outputs are stopped in a HIGH/LOW state, while the complementary (differential) output is stopped in a LOW/HIGH state. When OMODE is LOW and its corresponding bit 59 is 0, the outputs' disable state will be tri-state.

BYPASSING PLL

An option to bypass the PLL and operate in a clock buffer mode is provided by an external pin, $\overline{PLL\_EN}$, and its corresponding Bit 57. These two are used in conjunction and function as an NOR where both have to be LOW to enable the PLL. If either one is HIGH then the PLL will be disabled and the reference clock will be buffered. During the clock buffer mode (PLL disabled), all divide and skew selections will remain in effect, but the skew will still be based off the VCO frequency which would be idling at the minimum of its range. The idle frequency may vary and is suggested to not depend or use the skew time units for system timing adjustment.

POWERUP CONDITIONS

The parts will powerup with all internal volatile programming registers defaulted to a value of '0', bits 95:0. Upon powerup, the device will perform an automatic restore to load the contents of the EEPROM onto the internal programming registers. Initial factory shipment will have the EEPROM pre-loaded with all Bits set to '0'. The user must have the $\overline{PD}$ pin set HIGH to ensure proper operation of the auto-restore. The auto-restore will function regardless of the state of the $\overline{PLL\_EN}$ pin or Bit 57. Once the device acknowledges the 7-bit I²C address, it will be ready to accept a programming instruction. Assuming there is a valid clock on REF and FB, the total time, including complete power-up of the device, stabilization, auto-restore time and lock time of the PLL, is approximately 4ms. The auto-restore time itself will be approximately 3ms.

PROGRAMMABLE SKEW CAPABILITIES

The programmable skew PLLs offer up to 7 time units, lead or lag, per output bank, including the feedback bank. A nominal time unit is 1/16 of the VCO period. For example, if your reference clock frequency is 10MHz and the PLL is configured to multiply by 10 to achieve 100MHz on the outputs, then the VCO will be running at 100MHz. So the VCO period would be 10ns and 1/16 of that would yield 625ps for a nominal time unit. The maximum skew achieved would be ±(7*625ps) or ±4.375ns.

PROGRAMMABLE SKEW RANGE AND RESOLUTION TABLE

	FS = LOW	FS = HIGH	Comments
Timing Unit Calculation (t_U)	1/(16 x F_{NOM})	1/(16 x F_{NOM})	
VCO Frequency Range (F_{NOM})	50 to 125MHz	100 to 250MHz	
Skew Adjustment Range Max Adjustment:	±8.75ns	±4.375ns	ns
	±157.5°	±157.5°	Phase Degrees
	±43.75%	±43.75%	% of Cycle Time
Example 1, F_{NOM} = 50MHz	t_U = 1.25ns	—	
Example 2, F_{NOM} = 75MHz	t_U = 0.833ns	—	
Example 3, F_{NOM} = 100MHz	t_U = 0.625ns	t_U = 0.625ns	
Example 4, F_{NOM} = 150MHz	—	t_U = 0.417ns	
Example 5, F_{NOM} = 200MHz	—	t_U = 0.313ns	
Example 6, F_{NOM} = 250MHz	—	t_U = 0.25ns	

FREQUENCY MULTIPLICATION AND DIVISION

Each output bank has its own divide-by-2 and divide-by-4 to choose from. The configuration Bits 29:0 are in sets of five bits per each output bank, including the feedback bank. The table below shows the divide-by-2 and divide-by-4 settings.

JTAG/ I^2C SERIAL CONFIGURATIONS: SKEW OR FREQUENCY SELECT

Bit 4, 9, 14, 19, 24, 29	Bit 3, 8, 13, 18, 23, 28	Bit 2, 7, 12, 17, 22, 27	Bit 1, 6, 11, 16, 21, 26	Bit 0, 5, 10, 15, 20, 25	Output Skew
0	0	0	0	0	Zero Skew
1	0	0	0	0	Inverted
1	0	0	0	1	Divide-by-2
1	0	0	1	0	Divide-by-4

Configuration Bits 51:48 set the values for the internal FB divider to achieve clock multiplication. The available feedback divide ratios and settings are shown in the table below. The total feedback divide ratio (FB divider + post divider) is limited to a maximum of 12. For example, the FB divider can be set to /6 and the feedback bank can be set to /2 for an effective /12 feedback divide ratio. All four part numbers will have feedback divider options available.

JTAG/ I²C SERIAL CONFIGURATIONS: FB DIVIDE-BY-N

Bit 51	Bit 50	Bit 49	Bit 48	Divide-by-N	Permitted Output Divide-by-N connected to FB and $\overline{FB}$/VREF2
0	0	0	0	1	1, 2, 4
0	0	0	1	2	1, 2
0	0	1	0	3	1
0	0	1	1	4	1, 2
0	1	0	0	5	1, 2
0	1	0	1	6	1, 2
0	1	1	0	8	1
0	1	1	1	10	1
1	0	0	0	12	1

The allowable input frequency range and output frequency range are determined by the minimum and maximum VCO frequency and highest feedback divide ratio allowed. The minimum VCO frequency is 50MHz but the minimum input frequency allowed is 4.17MHz, based on a /12 feedback divide ratio, which would multiply the input clock by 12 to effectively achieve 50MHz in the VCO. The maximum input frequency is just based off the maximum VCO frequency of 250MHz.

The output frequency range is calculated similar to the input frequency range but instead of basing it on the feedback divide ratio, it is based off the post divide ratio allowed, which would be 4. So the minimum output frequency would be 12.5MHz based on the minimum VCO frequency of 50MHz and maximum is 250MHz.

INPUT AND OUTPUT I/O STANDARDS

All four devices offer four distinct I/Os on the outputs, 2.5V LVTTL, 1.8V LVTTL, 1.8V HSTL (eHSTL), and 1.5V HSTL. The I/O selection for each output bank is controlled via I²C/JTAG serial programming, which would be Bits 47-36. The dedicated V_{DDQ} pins must be connected appropriately for each I/O bank.

JTAG/ I²C SERIAL CONFIGURATIONS: OUTPUT DRIVE STRENGTH SELECTION

Bit 37, 39, 41, 43, 45, 47	Bit 36, 38, 40, 42, 44, 46	Interface
0	0	2.5V LVTTL
0	1	1.8V LVTTL
1	0	HSTL/eHSTL

On the input side, reference clock inputs support pure single-ended 3.3V LVTTL, 2.5V LVTTL, and 1.8V LVTTL and differential inputs such as LVPECL, LVEPECL, HSTL, eHSTL, LVDS, etc. The differential I/Os must meet the minimum input differential voltage of 400mV and common mode range of 600mV to 1400mV. The type of I/O the clock inputs will interface with is controlled via I²C/JTAG serial programming, which would be Bits 36-30. There are three distinct types of I/O input interfacing to select from, as shown in the table below. One selection would be for single-ended 2.5V LVTTL interfacing which also includes 3.3V LVTTL since the clock inputs are 3.3V tolerant. A second selection would be for single-ended1.8V LVTTL interfacing and the last is for differential I/Os. In Differential input interfacing mode, the input buffer can also accept single-ended I/Os such as LVTTL, as long as a reference voltage is applied to the complementary input.

JTAG/ I²C SERIAL CONFIGURATIONS: CLOCK INPUT INTERFACE SELECTION

Bit 31, 33, 35	Bit 30, 32, 34	Interface
0	0	Differential
0	1	2.5V LVTTL
1	1	1.8V LVTTL

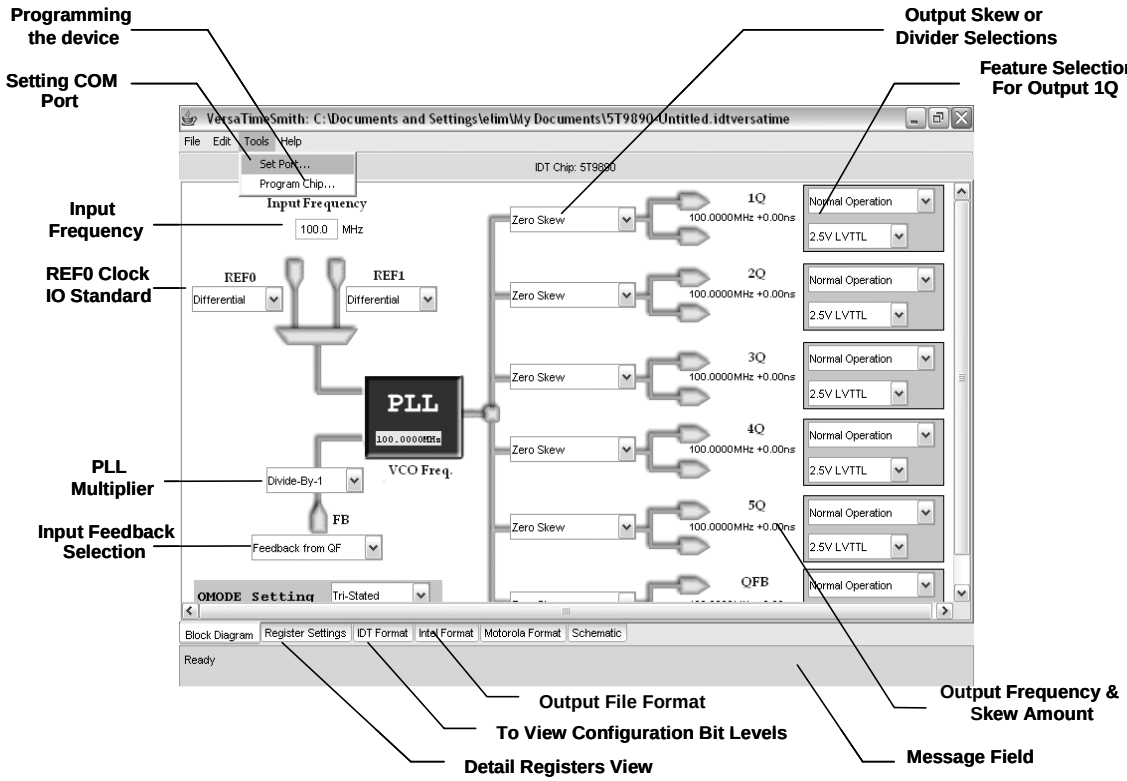
All the control pins accept 2.5V LVTTL and can accept 1.8V LVTTL by setting Bit 61.

USING THE PROGRAMMABLE CLOCK SOFTWARE AND PROGRAMMING KIT

IDT provides Programmable Clock software to configure the device and its features easily. Combined with the Timing Device Programmer kit, the software also allows users to program and test the device in the lab environment. The Programmable Clock software is included in the Timing Device Programmer kit, and can also be downloaded from the IDT website.

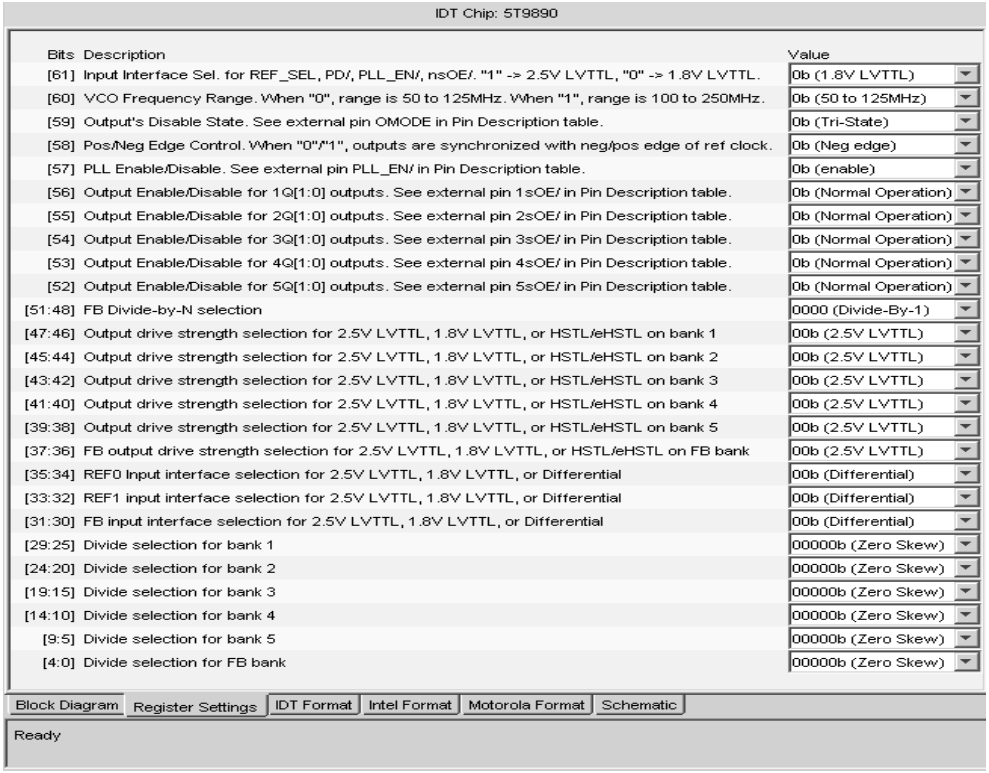
PROGRAMMABLE CLOCK SOFTWARE OVERVIEW

The Programmable Clock software provides six tab-screens to view all the features and bits: Block Diagram tab, Register Settings tab, IDT Format tab, Intel Format tab, Motorola Format tab and Schematics tab. The Block Diagram and Register Settings tab allow users to change the configurations easily, where the IDT Format, the Intel Format and the Motorola Format allows users to view their configuration bits in their respective formats. The Schematics tab shows the high level schematics diagram of the device.



In the Block Diagram tab, input frequency box is the only box that requires a user-entered value. The rest are pull-down menu, and based on user's selection on the pull-down menu, the software will update the results and bits automatically. Users can change input reference clock I/O standard, feedback divider values, output dividers and skew selections, output I/O standards, Tri-level selection and feedback input selection. The software will calculate the VCO frequency, the output frequencies and their corresponding skew values. Any changes in the Block Diagram tab will change the configuration bits automatically, which will also be reflected in the Register Settings tab. For instance, the software will automatically calculate the VCO frequency based on the input frequency value and feedback divider value, and then updates the VCO tuning range setting bit in the Register Setting tab.

The register settings can be changed to the desired configuration using either the block diagram or the register settings tab. Note that if using the block diagram, Bits 30, 31, 57, 58, and 61 must be set in the register settings tab.



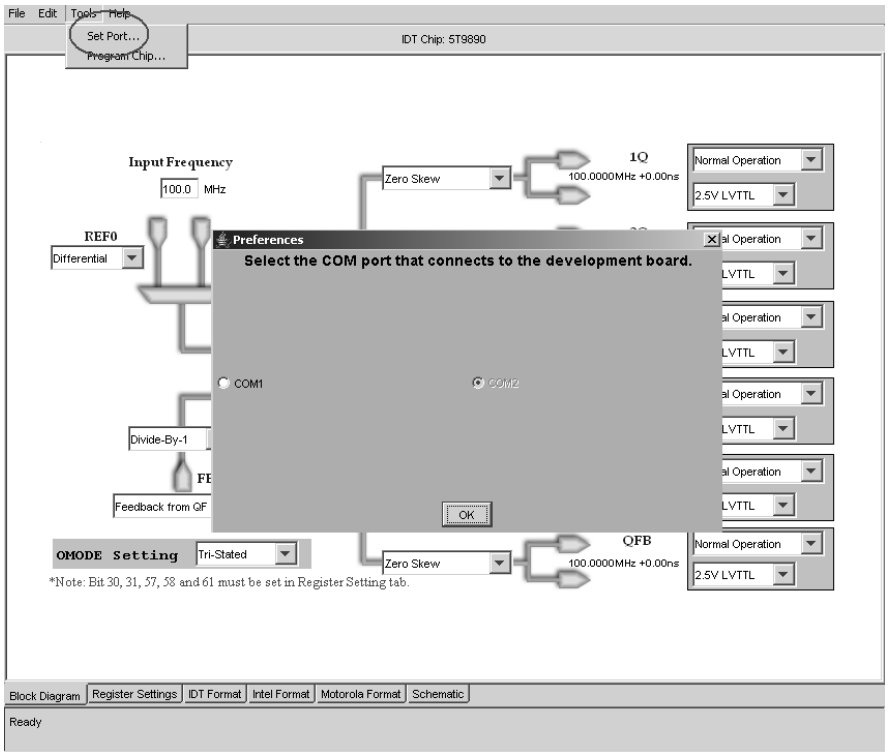
The Programmable Clock Software currently supports three file formats; IDT Format, Intel Hex Format, and Motorola S-record Format. The IDT Format is used to program the device when using Timing Device Programming kit. The file also includes more detail info about the configuration bits, as well as the time when the configuration file was created. Once the IDT Format file is saved, users can export the file into Intel Hex and Motorola S-record format. These formats are used to program the device through the JTAG interface. Most high end automatic programming equipment can input these file formats and program the device in a relatively straightforward manner.

PROGRAMMING WITH TIMING DEVICE PROGRAMMING KIT

To program this device using the Timing Device Programmer, Programmable Clock software is needed, and must be set up correctly.

Setting COM Port

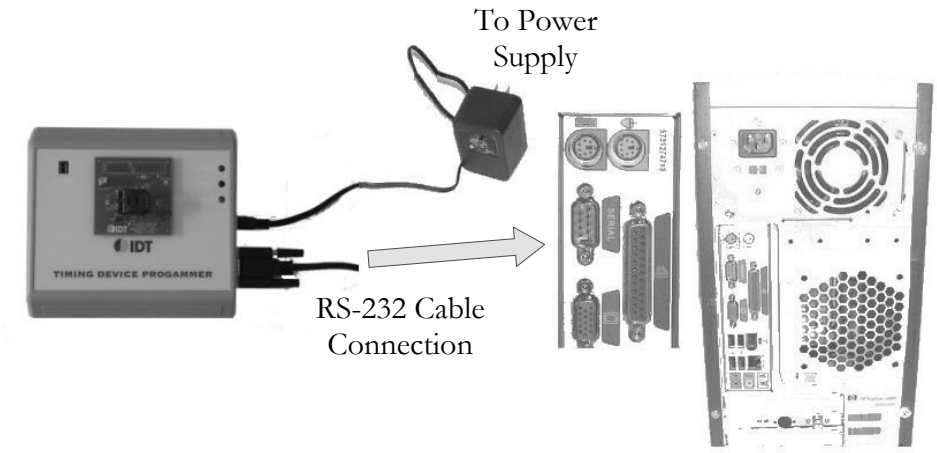
Once the features selections are done on the Block Diagram and Register Settings tabs, users will need to choose the appropriate COM port setting to program the device. To set the COM port, select Tools from the menu bar, and then select Set Port. Users can select the appropriate COM port setting.



Setting Up Hardware Connections

All hardware needs to be connected correctly, before programming is started. The module connector must be inserted into the Timing Device Programmer, and the correct device inserted into the socket.

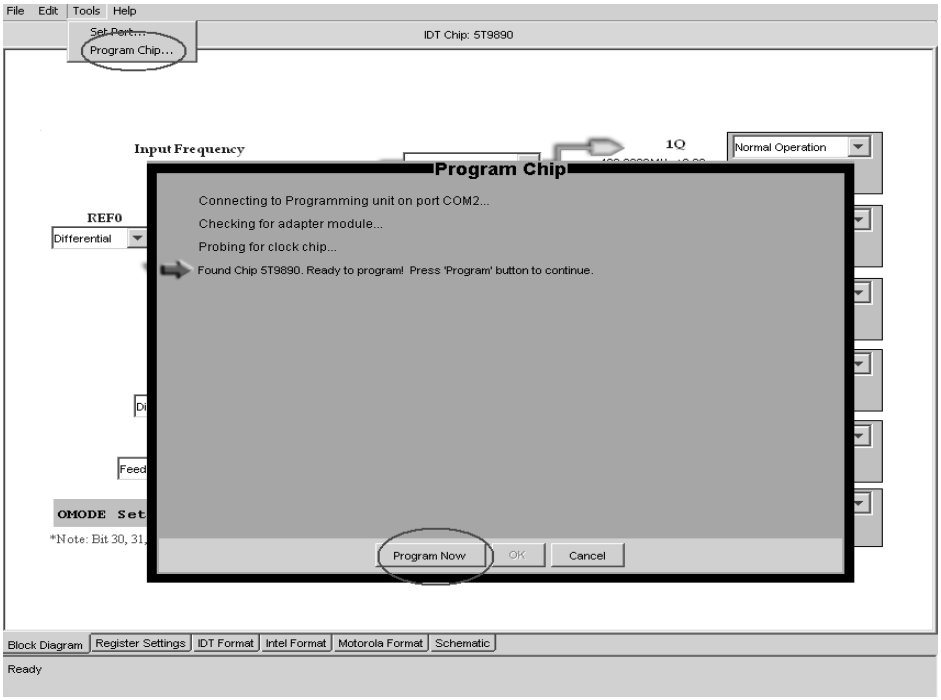
Connect the RS-232 and power supply to the Timing Device Programmer.



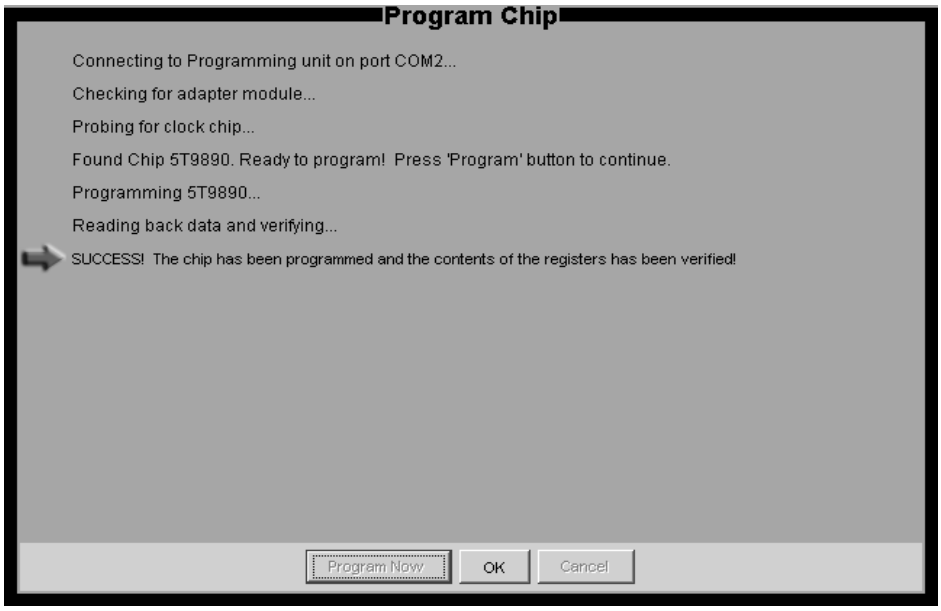
Start Programming

After the COM port setting selection and hardware setup, the device is ready to be programmed. Select Tools from the menu bar, and then select Program Chip.

The software will detect if the correct module is plugged-in or not, and then it will check if the device inside the socket matches the device ID from the configuration file. After successfully passing all these internal checks, the device is ready to be programmed.



Click "Program Now" to program the device. Unlike other programmable devices, these devices use EEPROM technology to store the configuration, and its EEPROM can be read/written to more than 100 times.



Error Messages

The Programmable Clock device register setting values are all preset to 0b as a default state. Since the internal EEPROM clock is based off the VCO of the PLL, it is important to have the PLL enabled when using the software. That means Bit 57 must be always set to 0b (PLL enabled). There will be no harm done to the device if Bit 57 is set to 1b, but it will not be properly programmed, which will be indicated by an error message as shown below.

If the socket lid is not properly locked, the device is not properly inserted (see pin 1 on the module PCB), or an incorrect clock device is used, then the software will prompt an error message as shown below.

Programming Summary (Step-by-Step)

Below is the step-by-step procedure to program the device using the Programmable Clock software and Timing Device Programming kit.

1. Launch the Programmable Clock Software application
2. Select the device to be programmed or configured
3. Select all the feature settings through Block Diagram tab and/or Register Settings tab
4. Save the configuration
5. Connect the Timing Device Programmer kit (see TDP Getting Started Guide)
6. Insert the Device Module into the Timing Device Programmer kit
7. Insert the correct device into the Device Module socket
8. Connect power supply and RS232 cable
9. From the software menu bar, select Tools -> Set Port to select the correct COM port setting
10. Select Tools -> Program Chip to program the device
    (The software will detect the connection, module, and device to make sure the device and the module matches the configuration file)
11. Then click Program Now to program the device
12. To program through Automatic Programming Equipment or JTAG programming software, Intel Hex or Motorola S-record file format may be needed. To export to these files,Go to File -> Export -> select Intel Hex or Motorola S-record

Prototype Board Programming Using the Timing Device Programmer

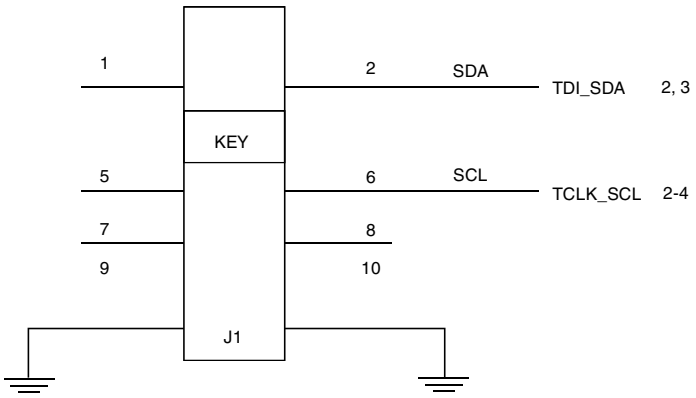
The Timing Device Programmer can be used to program the user's prototype board on the fly for testing purposes. The user needs to bring the I^2C pins of the device to a header, or to the I^2C connector listed below. After the Device Programmer is set up according to the procedure in the "Setting Up Hardware Connection" section, the user can connect the IDT I^2C cable from the I^2C external port on the Timing Device Programmer to the prototype board (see Timing Device Programmer Overview diagram below). Since Programmable Clock software detects for the module ID, it is important that the proper Socket Module (without the device inside) should be plugged into the programmer, even when the programming is through an I^2C cable.

The I^2C cable comes with the Device Programmer. It can also be requested from IDT with the IDT part number below. The I^2C connector used for I^2C interface is made by Samtec, the part number of which is also shown below. From there, the user can use IDT software to directly control and program the device on the prototype board.

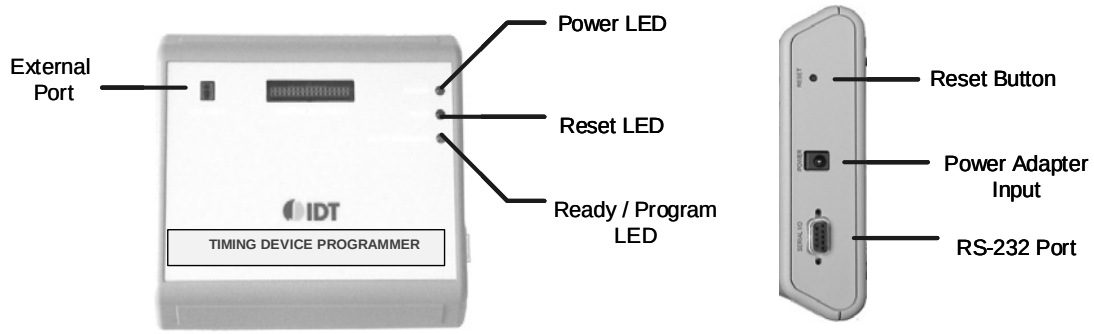
Cable Part Number: 52-534-000

Connector Part Number: MEC1-105-02-S-D-LC (CONN,SM,2 x 5,Edge-Card Skt,1.0mm PT,STR)

Drawing: **http://www.samtec.com/ftppub/cpdf/MEC1-1XX-XX-X-D-XX-XX-MKT.pdf**



Connector Drawing



Timing Device Programmer Overview

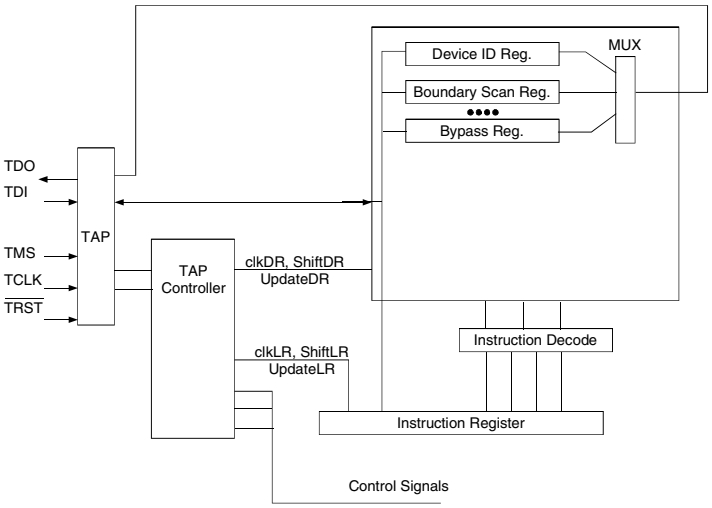
IN-SYSTEM PROGRAMMING

There are five pins ($\overline{TRST}$/SEL, TDO/ADDR1, TDI/ADDR0, TCK/SCLK, TDI/SDA) that have dual functionality, meaning that these pins can be used for both JTAG and I²C serial programming. The serial interface programming of choice will be determined by the $\overline{TRST}$/SEL pin. When the $\overline{TRST}$/SEL pin is HIGH, JTAG is enabled for serial programming. When LOW, I²C is enabled for serial programming while the JTAG boundary-scan circuitry is asynchronously reset.

Note that all the JTAG and I²C signal pins are only 2.5V tolerant.

IN-SYSTEM PROGRAMMING USING JTAG

The Programmable Clock devices can be programmed through JTAG interface. The devices have five JTAG pins (TDI, TDO, TMS, TCLK and $\overline{TRST}$) to support the JTAG boundary scan and JTAG programming. The devices also incorporate the necessary tap controller and modified pad cells to facilitate the JTAG capability. For a complete description refer to the IEEE Standard Test Access Port Specification (IEEE Std. 1149.1-1990).



Boundary Scan Architecture

The Standard JTAG interface consists of four basic elements:

- **Test Access Port (TAP)**
- **TAP controller**
- **Instruction Register (IR)**
- **Data Register Port (DR)**

Test Access Port (TAP)

The Tap interface is a general-purpose port that provides access to the devices internal. It consists of four input ports (TCLK, TMS, TDI, $\overline{TRST}$) and one output port (TDO).

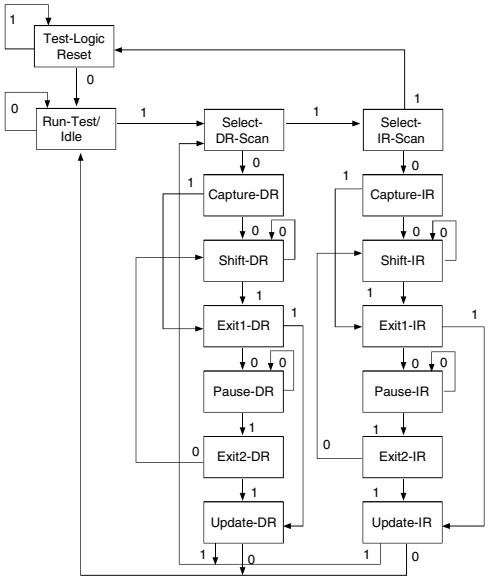
The Tap Controller

The Tap controller is a synchronous finite state machine that responds to TMS and TCLK signals to generate clock and control signals to the Instruction and Data Registers for capture and update of data.

Refer to the IEEE Standard Test Access Port Specification (IEEE Std.1149.1) for the full state diagram. All state transitions within the TAP controller occur at the rising edge of the TCLK pulse. The TMS signal level (0 or 1) determines the state progression that occurs on each TCLK rising edge. The TAP controller takes precedence over the PLL and must be reset after power up of the device. See $\overline{TRST}$ Pin description for more details on TAP controller reset.

Test-Logic-Reset

All test logic is disabled in this controller state enabling the normal operation of the IC. The TAP controller state machine is designed in such a way that, no matter what the initial state of the controller is, the Test-Logic-Reset state can be entered by holding TMS at high and pulsing TCLK five times. This is the reason why the Test Reset ($\overline{TRST}$) pin is optional.



JTAG TAP Controller State Diagram

NOTES:

1. Five consecutive TCLK cycles with TMS = 1 will reset the TAP.
2. TAP controller must be reset before normal PLL operations can begin.

Run-Test-Idle

In this controller state, the test logic in the IC is active only if certain instructions are present. For example, if an instruction activates the self test, then it will be executed when the controller enters this state. The test logic in the device is idle otherwise.

Select-IR-Scan

This is a controller state where the decision to enter the Instruction Path is made. The Controller can return to the Test-Logic-Reset state otherwise.

Capture-IR

In this controller state, the shift register bank in the Instruction Register parallel loads a pattern of fixed values on the rising edge of TCLK. The last two significant bits must be "01".

Shift-IR

In this controller state, the instruction register gets connected between TDI and TDO, and the captured pattern gets shifted on each rising edge of TCLK. The instruction available on the TDI pin is also shifted in to the instruction register.

Exit1-IR

This is a controller state where a decision to enter either the Pause-IR state or Update-IR state is made.

Pause-IR

This state is provided in order to allow the shifting of instruction register to be temporarily halted.

Update-IR

In this controller state, the instruction in the instruction register is latched in to the latch bank of the Instruction Register on every falling edge of TCLK. This instruction also becomes the current instruction once it is latched.

Capture-DR

In this controller state, the data is parallel loaded in to the data registers selected by the current instruction on the rising edge of TCLK.

Shift-DR, Exit1-DR, Pause-DR, Exit2-DR and **Update-DR**

These controller states are similar to the Shift-IR, Exit1-IR, Pause-IR, Exit2-IR and Update-IR states in the Instruction path.

Exit2-DR

This is a controller state where a decision to enter either the Shift-IR state or Update-IR state is made.

Select-DR-Scan

This is a controller state where the decision to enter the Data Path or the Select-IR-Scan state is made.

INSTRUCTION REGISTER (IR)

The Instruction register allows instruction to be serially input into the device when the TAP controller is in the Shift-IR state. The instruction is decoded to perform the following:

Select test data registers that may operate while the instruction is current. The other test data registers should not interfere with chip operation and the selected data register.

Define the serial test data register path that is used to shift data between TDI and TDO during data register scanning. The Instruction Register is a 4-bit field (i.e. IR3, IR2, IR1, IR0) to decode 16 different possible instructions. Instructions are decoded in the following table:

JTAG INSTRUCTION REGISTER DECODING

IR (3)	IR (2)	IR (1)	IR (0)	Instruction	Function
0	0	0	0	EXTEST	Select boundary scan register
0	0	0	1	SAMPLE/PRELOAD	Select boundary scan register
0	0	1	0	IDCODE	Select chip identification data register
0	0	1	1		Reserved
0	1	0	0	PROGWRITE	Writing to the volatile programming registers
0	1	0	1	PROGREAD	Reading from the volatile programming registers
0	1	1	0	PROGSAVE	Saving the contents of the volatile programming registers to the EEPROM
0	1	1	1	PROGRESTORE	Loading the EEPROM contents into the volatile programming registers
1	0	0	0	CLAMP	JTAG
1	0	0	1	HIGHZ	JTAG
1	0	1	X	BYPASS	Select bypass register
1	1	X	X	BYPASS	Select bypass register

The following sections provide a brief description of each instruction. For a complete description refer to the IEEE Standard Test Access Port Specification (IEEE Std. 1149.1-1990).

EXTEST

The required EXTEST instruction places the IC into an external boundary test mode and selects the boundary-scan register to be connected between TDI and TDO. During this instruction, the boundary-scan register is accessed to drive test data off-chip through the boundary outputs, and recieve test data off-chip through the boundary inputs. As such, the EXTEST instruction is the workhorse of IEEE. Std 1149.1, providing for probe-less testing of solder-joint opens/shorts and of logic cluster function.

SAMPLE/PRELOAD

The required SAMPLE/PRELOAD instruction allows the IC to remain in a normal functional mode and selects the boundary-scan register to be connected between TDI and TDO. During this instruction, the boundary-scan register can be accessed via a data scan operation, to take a sample of the functional data entering and leaving the IC.

IDCODE

The optional IDCODE instruction allows the IC to remain in its functional mode and selects the optional device identification register to be connected between TDI and TDO. The device identification register is a 32-bit shift register containing information regarding the IC manufacturer, device type, and version code. Accessing the device identification register does not interfere with the operation of the IC. Also, access to the device identification register should be immediately available, via a TAP data-scan operation, after power up or after the TAP has been reset using the optional $\overline{\text{TRST}}$ pin or by otherwise moving to the Test-Logic-Reset state.

PROGWRITE

The PROGWRITE instruction is for writing the device configuration data to the device's volatile programming registers. This instruction selects the programming register path for shifting data from TDI to TDO during data register scanning. The programming register path has 112 registers (14 bytes) between TDI and TDO. The 12 configuration data bytes are scanned in through TDI first, starting with Bit 0. After scanning in the last configuration bit, Bit 95, 16 additional bits must be scanned in to place the configuration data in the proper location. The last 16 registers in the programming path are reserved, read-only registers.

PROGREAD

The PROGREAD instruction is for reading out the device configuration data from the device's volatile programming registers. This instruction selects the programming register path for shifting data from TDI to TDO during data register scanning. The programming register path has 112 registers between TDI and TDO, and the first bit scanned out through TDO will be Bit 0 of the configuration data.

PROGSAVE and **PROGRESTORE (EEPROM OPERATION)**

The PROGSAVE instruction is for copying the device configuration data from the device's volatile programming registers to the EEPROM. This instruction selects the BYPASS register path for shifting data from TDI to TDO during data register scanning.

The PROGRESTORE instruction is for loading the device configuration data from the EEPROM to the device's volatile programming registers. This instruction selects the BYPASS register path for shifting data from TDI to TDO during data register scanning. During the execution of a PROGSAVE or PROGRESTORE instruction, the device will not accept a new programming instruction (read, write, save, or restore). All non-programming JTAG instructions will function properly, but the user should wait until the save or restore is complete before issuing a new programming instruction. The time it takes for the save and restore instructions to complete depends on the PLL oscillator frequency, FVCO. The restore time, TRESTORE, and the save time, TSAVE, can be calculated as follows:

$TRESTORE = 1.23 \times 10^6/FVCO\ (mS)$
$TSAVE = 3.09 \times 10^6/FVCO + 52\ (mS)$

If a new programming instruction is issued before the save or restore completes, the new instruction is ignored, and the BYPASS register path remains in effect for shifting data from TDI to TDO during data register scanning. In order for the ProgSave and ProgRestore instructions to function properly, the device must not be in power-down mode ($\overline{PD}$ must be HIGH), and the PLL must be enabled ($\overline{PLL\_EN}$ = LOW and Bit 57 = 0).

On power-up, an automatic restore is performed to load the EEPROM contents into the internal programming registers. The autorestore will not function properly if the device is in power-down mode ($\overline{PD}$ must be HIGH). The device's auto-restore feature will function regardless of the state of the $\overline{PLL\_EN}$ pin or Bit 57. The time it takes for the device to complete the auto-restore is approximately 3ms.

CLAMP

The optional CLAMP instruction loads the contents from the boundary-scan register onto the outputs of the IC, and selects the one-bit bypass register to be connected between TDI and TDO. During this instruction, data can be shifted through the bypass register from TDI to TDO without affecting the condition of the IC outputs.

HIGH-IMPEDANCE

The optional High-Impedance instruction sets all outputs (including two-state as well as three-state types) of an IC to a disabled (high-impedance) state and selects the one-bit bypass register to be connected between TDI and TDO. During this instruction, data can be shifted through the bypass register from TDI to TDO without affecting the condition of the IC outputs.

BYPASS

The required BYPASS instruction allows the IC to remain in a normal functional mode and selects the one-bit bypass register to be connected between TDI and TDO. The BYPASS instruction allows serial data to be transferred through the IC from TDI to TDO without affecting the operation of the IC.

DATA REGISTER (DR)

The test Data Register contains three test data registers: the Bypass, the Boundary Scan register and Device ID register. These registers are connected in parallel between a common serial input

and a common serial data output. The following sections provide a brief description of each element. For a complete description, refer to the IEEE Standard Test Access Port Specification

(IEEE Std. 1149.1-1990).

Test Bypass Register

The register is used to allow test data to flow through the device from TDI to TDO. It contains a single stage shift register for a minimum length in serial path. When the bypass register is selected by an instruction, the shift register stage is set to a logic zero on the rising edge of TCLK when the TAP controller is in the Capture-DR state. The operation of the bypass register should not have any effect on the operation of the device in response to the BYPASS instruction.

Test Boundary Scan Register

The Boundary Scan Register allows serial data TDI be loaded in to or read out of the processor input/output ports. The Boundary Scan Register is a part of the IEEE 1149.1-1990 Standard JTAG Implementation.

Device ID Register

The Device Identification Register is a Read Only 32-bit register used to specify the manufacturer, part number and version of the processor to be determined through the TAP in response to the IDCODE instruction. IDT JEDEC ID number is 0xB3. This translates to 0x33 when the parity is

dropped in the 11-bit Manufacturer ID field. For the IDT5T9890, the Part Number field is 0x3A8.

Device Name	JTAG Part Number
5T9890	0x3A8
5T9891	0x3A6
5T9820	0x3A9
5T9821	0x3A7

JTAG DEVICE IDENTIFICATION REGISTER

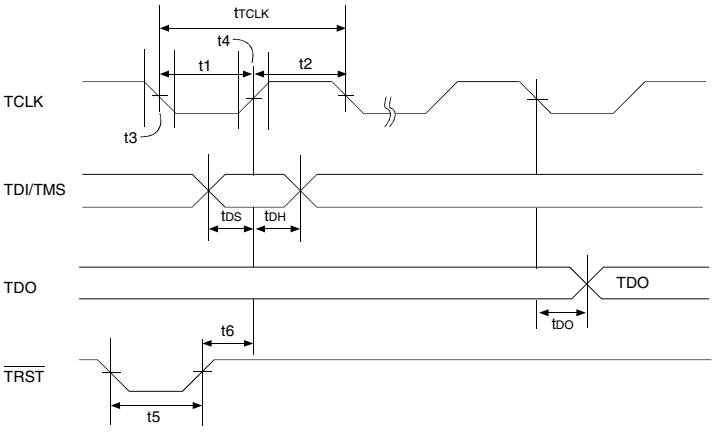
31 (MSB) 28	27 12	11 1	0(LSB)
Version (4 bits) 0X0	Part number (16-bit)	Manufacturer ID (11-bit) 0X33	1

JTAG PROGRAMMING NOTES

Once the device has been programmed either with a ProgWrite or ProgRestore instruction, it will attempt to achieve phase lock using the new PLL configuration. The Programmable Clock Software can export the configuration information into Intel Hex and Motorola S record files formats. These files can be used directly with most of the JTAG programming or boundary scanning software. If there is a valif REF and FB input clock connected to the device, and it does not achieve lock, the user should issue a ProgRead instruction to confirm that the PLL configuration data is valid.

On power-up and before the automatic ProgRestore instruction has completed, the internal programming registers will contain the value of '0' for all bits 95:0. The PLL will remain at the minimum frequency and will not achieve phase lock until after the automatic restore is completed. If the outputs are enabled by the $\overline{nSOE}$ pins, the outputs will toggle at the minimum frequency. If the outputs are disabled by the $\overline{nSOE}$ pins, and the OMODE pin is set high, the nQ[1:0] and QFB are stopped HIGH, while $\overline{QFB}$ is stopped LOW.

Please see the device datasheet for detailed JTAG timing information.



NOTE:
t1 = $t_{TCLKLOW}$
t2 = $t_{TCLKHIGH}$
t3 = $t_{TCLKFALL}$
t4 = $t_{TCLKRISE}$
t5 = t_{RST} (reset pulse width)
t6 = t_{RSR} (reset recovery)

IN-SYSTEM PROGRAMMING USING I²C

The Programmable Clock devices support I²C interface and meet Philips I²C bus specifications. The I²C bus is controlled by an external master device that generates the serial clock SCLK and commands through SDA. Both master and slave can operate as a transmitter and receiver but the master device determines which mode is activated.

BUS CONDITIONS

Data transfer on the bus can only be initiated when the bus is not busy. During data transfer, the data line (SDA) must remain stable whenever the clock line (SCLK) is high. Changes in the data line while the clock line is high will be interpreted by the device as a START or STOP condition. The following bus operating conditions are defined by the I²C bus protocol and are illustrated in figure 1.

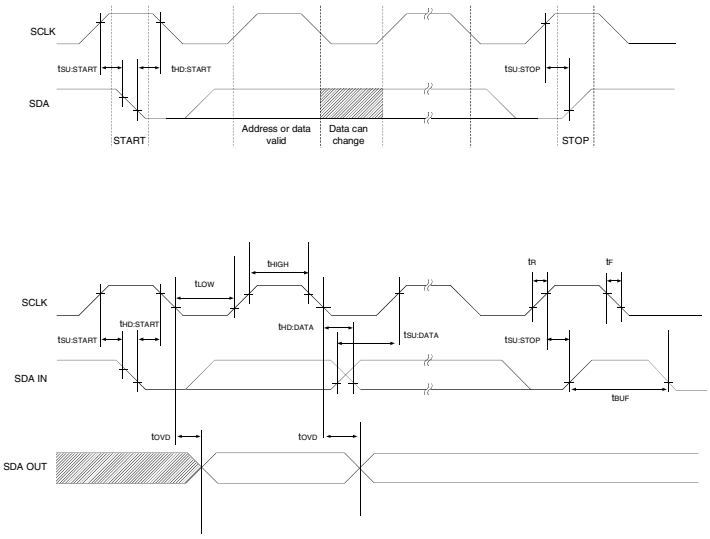


Figure 1: I²C Timing Data

NOT BUSY condition

Both the data (SDA) and clock (SCLK) lines remain high to indicate the bus is not busy.

START Data Transfer condition

A high to low transition of the SDA line while the SCLK input is high indicates a START condition. All commands to the device must be preceded by a START condition.

STOP Data Transfer condition

A low to high transition of the SDA line while SCLK is held high indicates a STOP condition. All commands to the device must be followed by a STOP condition.

DATA VALID condition

The state of the SDA line represents valid data if the SDA line is stable for the duration of the high period of the SCLK line after a START condition occurs. The data on the SDA line must be hanged only during the low period of the SCLK signal. There is one clock pulse per data bit. Each data transfer is initiated by a START condition and terminated with a STOP condition.

ACKNOWLEDGE condition

When addressed, the receiving device is required to generate an ACKNOWLEDGE after each byte is received. The master device must generate an extra clock pulse to coincide with the Acknowledge bit. The acknowledging device must pull the SDA line low during the high period of the master acknowledge clock pulse. Setup and hold times must be taken into account.

I²C BUS OPERATION

The Programmable Clock devices support Standard-Mode (100kHz) and Fast-Mode (400kHz) data transfer rates for I²C interface. Data is transferred in bytes in sequential order from the lowest to highest byte. After generating a START condition, the bus master broadcasts a 7-bit slave address followed by a read/ write bit. Each byte is followed by ACKNOWLEDGE from the receiving device. After ACKNOWLEDGE condition, the master device sends the command to determine the I²C operation.

A0 is the read/write bit and is set to '0' for writes and '1' for reads. To avoid addressing conflict when multiple Programmable Clock devices are on the same I²C bus, they provide two tri-level pins (ADDR0 and ADDR1 pins) to define the values of A1, A2 and A3 bits. Users can use these two pins to set different addresses on the bus.

I²C ADDRESS

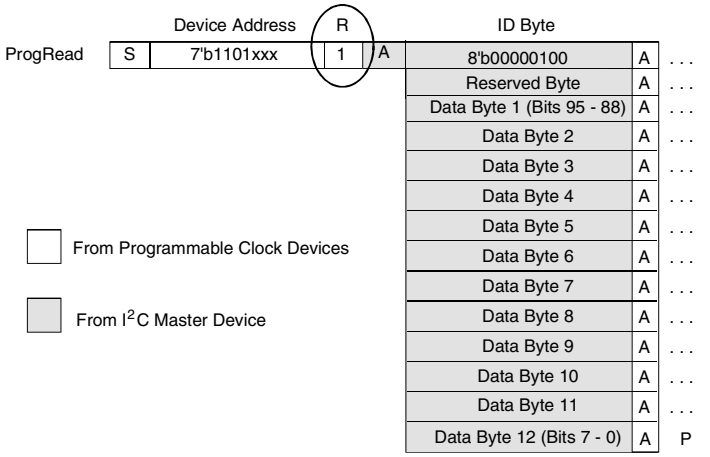
A7	A6	A5	A4	A3	A2	A1
1	1	0	1	X	X	X

ADDR1	ADDR0	A3	A2	A1
LOW	LOW	0	0	0
LOW	MID	0	0	1
LOW	HIGH	0	1	0
MID	LOW	0	1	1
MID	MID	1	0	0
MID	HIGH	1	0	1
HIGH	LOW	1	1	0
HIGH	MID	1	1	1
HIGH	HIGH	1	1	0

READ Operation (ProgRead command)

To read the PLL and device register bits values, ProgRead command is issued. The READ operation (or ProgRead command) is determined by setting read/write bit (A0) to '1'. During the read operation, there will be a total of 14 data bytes returned after an ACKNOWLEDGE from the Programmable Clock devices. The I²C master device must issue ACKNOWLEDGE upon every byte received. The first two data bytes are the device ID byte followed by Reserved byte. The subsequent bytes are the same 12 data bytes (PLL registers values) that were written during the write operation. The READ operation can be terminated at any time by issuing a STOP condition.

	I²C Device ID Byte							
Device	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
5T9890	0	0	0	0	0	1	1	0
5T9891	0	0	0	0	0	1	1	1
5T9820	0	0	0	0	0	1	0	0
5T9821	0	0	0	0	0	1	0	1



WRITE Operation (ProgWrite command)

To write to the PLL and device registers, ProgWrite command is issued. The WRITE operation (or ProgWrite command) is initiated by setting read/write bit (A0) to '0'. During the write operation, the I²C master device issues START condition and DATA VALID condition, followed by the device address, followed by ProgWrite command code (8'bxxxxxx00). The twelve Data Bytes are transferred after that, followed by STOP condition to complete the WRITE operation. If there is any interruption on the bus or STOP condition issued during the data bytes transfer, the internal programming registers will remain unchanged to prevent an invalid PLL configuration. An ACKNOWLEDGE condition from the Programmable Clock device must be issued upon receiving each byte. Once the STOP condition has occurred and all twelve data bytes are received, the internal programming registers will be updated.

		Device Address	W		Command Code		Data		
ProgWrite	S	7'b1101xxx	0	A	8'bxxxxxx00	A	Data Byte 1 (Bits 95 - 88)	A	...
		MSB LSB			MSB LSB		Data Byte 2	A	...
							Data Byte 3	A	...
							Data Byte 4	A	...
							Data Byte 5	A	...
							Data Byte 6	A	...
							Data Byte 7	A	...
							Data Byte 8	A	...
							Data Byte 9	A	...
							Data Byte 10	A	...
							Data Byte 11	A	...
							Data Byte 12 (Bits 7 - 0)	A	P

EEPROM WRITE Operation (ProgSave command)

Thes devices can also save the contents of the device configuration registers in internal EEPROM by issuing the ProgSave command. To initiate the ProgSave command, the I²C master device issues START and DATA VALID conditions, followed by device address and WRITE commands, and then followed by ProgSave command code (8'bxxxxxx01).

		Device Address	W		Command Code		
ProgSave	S	7'b1101xxx	0	A	8'bxxxxxx01	A	P

The EEPROM WRITE time depends on the PLL VCO frequency, and it can be calculated as follows:

$$TSAVE = 3.09 \times 10^6/FVCO + 52 \text{ (mS)}$$

EEPROM READ Operation (ProgRestore command)

Users can read the configuration register values inside the EEPROM by issuing ProgRestore commands, followed by READ operations. ProgRestore will restore the configuration values inside the EEPROM into device configuration registers, and READ operation (or ProgRead) will retreat the values of configuration registers to the I²C master device. To initiate ProgRestore, the master must issue START and DATAVALID conditions, followed by the device address and WRITE bit (A0 to be '0'), and then the ProgRestore command.

		Device Address	W		Command Code		
ProgRestore	S	7'b1101xxx	0	A	8'bxxxxxx10	A	P

The EEPROM READ operation (ProgRestore) time depends on the PLL VCO frequency, and it can be calculated as follows:

$$TRESTORE = 1.23 \times 10^6/FVCO \text{ (mS)}$$

I²C PROGRAMMING NOTES

The device must not be in power-down mode (PD must be HIGH), and the PLL must be enabled ($\overline{PLL\_EN}$ must be LOW and Bit 57 = 0) for any EEPROM operation. During the power-up time, an automatic restore is performed to load the EEPROM contents into the internal programming registers. The auto-restore process takes approximately 3ms.

Once the device has been programmed either with a ProgWrite or ProgRestore instruction, the device will attempt to achieve phase lock using the new PLL configuration. If there is a valid REF and FB input clock connected to the device and it does not achieve lock, the user should issue a ProgRead instruction to confirm that the PLL configuration data is valid. On power-up and before the automatic ProgRestore instruction has completed, the internal programming registers will contain the value of '0' for all bits 95:0. The PLL will remain at the minimum frequency and will not achieve phase lock until after the automatic restore is completed. If the outputs are enabled by the $\overline{nSOE}$ pins, the outputs will toggle at the minimum frequency. If the outputs are disabled by the $\overline{nSOE}$ pins and the OMODE pin is set HIGH, the nQ[1:0] and QFB are stopped HIGH, while $\overline{QFB}$ is stopped LOW.

Please see the device datasheet for detailed I²C timing information.

DEVICE FEATURES PROGRAMMING EXAMPLES

APPLICATION REQUIREMENTS:

- **Zero delay buffer device**
- **Two reference inputs - one is 2.5V LVTTL and another one is LVPECL; both input frequencies are 50MHz**
- **Three outputs: 200MHz, 100MHz, and 50MHz with swek**
- **JTAG programming**

CONFIGURING THE PINS AND REGISTERS

The following section describes how to configure the device to meet the application requirements.

JTAG/I²C PIN CONFIGURATION

TRST/SEL is set HIGH for JTAG programming

Parameter	Value	Description
TRST/SEL	Set HIGH for JTAG	JTAG is enabled for programming
TDI/SDA	JTAG	JTAG serial data input
TCLK/SCLK	JTAG	JTAG clock for BST
TMS/ADDR0	JTAG	JTAG control signal
TDO/ADDR1	JTAG	JTAG serial data output

MISC. EXTERNAL PIN CONFIGURATION

Powerdown ($\overline{PD}$) pin is set HIGH for normal operation

Parameter	Value	Description
Powerdown ($\overline{PD}$)	set HIGH for Normal Operation	Device is fully operational
Reference Select (REF_SEL)	set HIGH for REF1	Selects REF1 clock at 50MHz
PLL Enable/Disable ($\overline{PLL\_EN}$)	See PLL Pin/Bit Configuration table	
Output Disable State (OMODE)	See Output Pin/Bit Configuration table	
Output Enable/Disable ($\overline{nsOE}$)	See Output Pin/Bit Configuration table	

INPUT BIT CONFIGURATION

Parameter		Bits	Register Setting	Value	Description
Reserved		95:62	0b	NA	Bits are all set to 0b
Control Pins Interface		61	1b	2.5V LVTTL	Accept 2.5V LVTTL signals
Input I/O Interface	REF0	35:34	01b	1.8V LVTTL	I/O is 2.5V LVTTL
	REF1	33:32	00b	Differential	I/O is LVPECL
	FBin	31:30	00b	Differential	I/O is HSTL

PLL PIN/BIT CONFIGURATION

Parameter		Bits	Register Setting	Value	Description
PLL Enable/Disable	Internal	57	0b	Enable	PLL enabled. Bit 57 and $\overline{PLL\_EN}$ must be 0b and LOW, respectively, to enable.
	External Pin, $\overline{PLL\_EN}$	NA	NA	set LOW for Enable	
PLL Positive/NegativeEdge Control		58	1b	Positive Edge	PLL is synched to positive edge of reference clock
FB Divider		51:48	0011b	4	Reference clock frequency is multiplied by 4
VCO frequency range		60	1b	100MH-250MHz	VCO frequency is at 200MHz (tu = 312.5ps)

OUTPUT PIN/BIT CONFIGURATION

Bank 3, 4 and 5 are used to generate the required output frequencies

Parameter			Bits	Register Setting	Value	Description
Output Disable State	Internal		59	0b	Tri-State	Output will tri-state when disabled. Bit 59 and OMODE must be 0b and LOW, respectively, for tri-state.
	External Pin, OMODE		NA	NA	set LOW for Tri-State	
Bank 1	Skew and Divide Selection		29:25	00001b	+7tu	200MHz with +7tu
	I/O interface		47:46	10b	eHSTL	I/O is eHSTL, $V_{DDQ}1$ = 1.8V
	Enable/Disable	Internal	56	0b	Enable	Output enabled. Bit 56 and $\overline{1sOE}$ must be 0b and LOW, respectively, to enable output
		External Pin, $\overline{1sOE}$	NA	NA	Set LOW for Enable	
Bank 2	Skew and Divide Selection		24:20	10000b	Invert	Inverted 200MHz
	I/O interface		45:44	00b	2.5V LVTTL	I/O is 2.5V LVTTL, $V_{DDQ}2$ = 2.5V
	Enable/Disable	Internal	55	1b	Disable	Output disabled. Bit 55 or $\overline{2sOE}$ must be 1b or HIGH, respectively, to disable output
		External Pin, $\overline{2sOE}$	NA	NA	Set LOW for Enable	
Bank 3	Skew and Divide Selection		19:15	10001b	Divide-by-2	100MHz output
	I/O interface		43:42	01b	1.8V LVTTL	I/O is 1.8V LVTT, $V_{DDQ}3$ = 1.8V
	Enable/Disable	Internal	54	0b	Enable	Output enabled. Bit 54 and $\overline{3sOE}$ must be 0b and LOW, respectively, to enable output
		External Pin, $\overline{3sOE}$	NA	NA	Set LOW for Enable	
Bank 4	Skew and Divide Selection		14:10	01011b	-3tu	200MHz with -3tu
	I/O interface		41:40	01b	1.8V LVTTL	I/O is 1.8V LVTTL, $V_{DDQ}4$ = 1.8V
	Enable/Disable	Internal	56	0b	Enable	Output enabled. Bit 53 and $\overline{4sOE}$ must be 0b and LOW, respectively, to enable output
		External Pin, $\overline{4sOE}$	NA	NA	Set LOW for Enable	
Bank 5	Skew and Divide Selection		9:5	10010b	Divide-by-4	50MHz output
	I/O interface		39:38	00b	2.5V LVTTL	I/O is 2.5V LVTTL, $V_{DDQ}5$ = 2.5V
	Enable/Disable	Internal	55	1b	Disable	Output disabled. Bit 52 or $\overline{5sOE}$ must be 1b or HIGH, respectively, to disable output
		External Pin, $\overline{5sOE}$	NA	NA	Set LOW for Enable	
Bank FBOUT	Skew and Divide Selection		4:0	0000b	Zero skew	200MHz feedback
	I/O interface		37:36	10b	HSTL	I/O is HSTL, V_{DDQFB} = 1.5V
	Enable/Disable		NA	NA	N/A	Cannot disable this bank

Below is the table for skew or frequency selection, using JTAG/I²C interface for reference.

SKEW OR FREQUENCY SELECTION

Bit 4, 9, 14, 19, 24, 29	Bit 3, 8, 13, 18, 23, 28	Bit 2, 7, 12, 17, 22, 27	Bit 1, 6, 11, 16, 21, 26	Bit 0, 5, 10, 15, 20, 25	Output Skew
0	0	0	0	1	+7tu
0	0	0	1	0	+6tu
0	0	0	1	1	+5tu
0	0	1	0	0	+4tu
0	0	1	0	1	+3tu
0	0	1	1	0	+2tu
0	0	1	1	1	+1tu
0	0	0	0	0	Zero Skew
0	1	0	0	1	-1tu
0	1	0	1	0	-2tu
0	1	0	1	1	-3tu
0	1	1	0	0	-4tu
0	1	1	0	1	-5tu
0	1	1	1	0	-6tu
0	1	1	1	1	-7tu
1	0	0	0	0	Inverted
1	0	0	0	1	Divide-by-2
1	0	0	1	0	Divide-by-4

USING PROGRAMMABLE I/O FEATURE

As mentioned earlier, all four devices can accept single-ended 3.3V, 2.5V, and 1.8V LVTTL and differential signals such as 1.8V/1.5V HSTL, LVPECL, 3.3V/2.5V/1.8V SSTL, and LVDS on the reference clock inputs. On the outputs, all four devices support 2.5 LVTTL, 1.8V LVTTL, 1.8V HSTL, and 1.5V HSTL. The Input/Output Selection table describes an example of the various combinations for each output bank.

INPUT/OUTPUT SELECTION(1)

Input	Output(2)
2.5V LVTTL SE	2.5V LVTTL,
1.8V LVTTL SE	1.8V LVTTL,
2.5V LVTTL DSE	HSTL,
1.8V LVTTL DSE	eHSTL
LVEPECL DSE	
eHSTL DSE	
HSTL DSE	
2.5V LVTTL DIF	
1.8V LVTTL DIF	
LVEPECL DIF	
eHSTL DIF	
HSTL DIF	

There are no internal terminations on the clocks inputs, high impedance. When interfacing with LVPECL, it is recommend to use AC coupling and a thevenin equivalent termination of 50Ω to V_{DD}-2.0V. A V_{DD} of 3.3V will yield about 1.3V bias voltage to the input buffer which would be acceptable for these devices.

For LVDS, DC coupling would be the best option since its common mode voltage is 1.2V which would be acceptable for the input buffer. SSTL I/Os would require a reference voltage of V_{DDQ}/2 on the complementary input. HSTL I/Os, if used as single-ended, would also require a reference of V_{DDQ}/2 on the complementary input.

All four output I/O interfaces are terminated with a 50Ω to V_{DDQ}/2 for device characterization but only HSTL would require this type of termination in an application since it is standard for HSTL Type-I, according to JESD8-06. Refer to AN-230 for more details on I/O standards.

V_{DD} and V_{SS} Connections

Each output bank has its own dedicated V_{DDQ} and can be set to the appropriate level depending on the I/O desired. Proper decoupling is recommend; a 0.1uF cap for each V_{DDQ} would be sufficient. V_{DD} is the power supply to all other internal circuitry such as the PLL and input buffers. A filter is recommended to isolate the V_{DD} and attenuate any noise from the power plane. All decoupling caps should be placed to the power pins as close as possible. When using 2.5V LVTTL outputs, the V_{DDQ} can be connected to V_{DD}.

IMPORTANT NOTICE AND DISCLAIMER

RENESAS ELECTRONICS CORPORATION AND ITS SUBSIDIARIES (“RENESAS”) PROVIDES TECHNICAL SPECIFICATIONS AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT OF THIRD-PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for developers who are designing with Renesas products. You are solely responsible for (1) selecting the appropriate products for your application, (2) designing, validating, and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. Renesas grants you permission to use these resources only to develop an application that uses Renesas products. Other reproduction or use of these resources is strictly prohibited. No license is granted to any other Renesas intellectual property or to any third-party intellectual property. Renesas disclaims responsibility for, and you will fully indemnify Renesas and its representatives against, any claims, damages, costs, losses, or liabilities arising from your use of these resources. Renesas' products are provided only subject to Renesas' Terms and Conditions of Sale or other applicable terms agreed to in writing. No use of any Renesas resources expands or otherwise alters any applicable warranties or warranty disclaimers for these products.

(Disclaimer Rev.1.01)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit www.renesas.com/contact-us/.

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.