XAPP198 (v1.0) May 8, 2001

Synthesizable FPGA Interface for Retrieving ROM Number from 1-Wire Devices

Author: Dai Huang and Rick Ballantyne

Summary

This application note describes the design and implementation of a simple, low-cost interface to the Dallas Semiconductor's 1-Wire™ devices in Virtex™ and Spartan™-II families to acquire the 64-bit ROM number. The number is available in either eight sequential byte transfers through an 8-bit data port, or a 48-bit latched parallel output. A typical application is to use the 48-bit serial number in the ROM number as the physical address of a network interface. This reference design is synthesizable and utilizes only 52 registers, 65 look-up tables (LUTs), and 55 slices of FPGA resource.

Introduction

As a member of Dallas Semiconductor's 1-Wire family, the DS2401 Silicon Serial Number enables unique network identification and addressability for the system with the device attached. The DS2401 device consists of a factory-lasered, 64-bit ROM that includes an 8-bit Family Code (`01h`), a unique 48-bit serial number, and an 8-bit Cyclic Redundancy Check (CRC) code, as shown in Figure 1. Data is transferred serially via the 1-Wire protocol, which requires only a single data lead and a ground return. Power for reading and writing to the device is derived from the data line itself. The 1-Wire protocol defines 1-Wire bus transactions in terms of the bus state during specified time slots.

This application note describes a 1-Wire device interface implemented in a Virtex or Spartan-II series FPGA. It is designed to be mapped into the user's system and provides control of the 1-Wire bus, in order to retrieve the 48-bit serial number from the 1-Wire device attached to the bus. It is easily implemented and operated from a spare FPGA port pin. Note that most devices (e.g., DS2430A EEPROM or DS1822 RAM) in the Dallas 1-Wire family contains the 64-bit ROM number, which can be retrieved through the same interface. In our application, we have tested this interface for both the DS2401 and DS2430A devices.

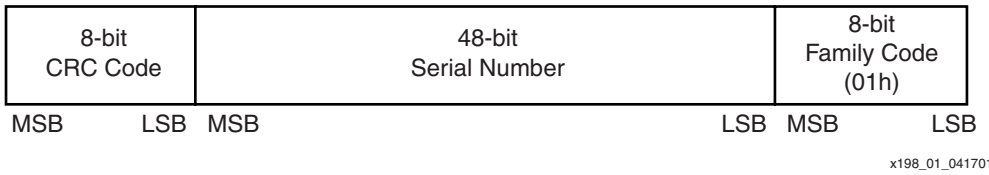

Figure 1: **DS2401 64-Bit ROM MAP**

A typical application using the DS2401 device is to store the physical address for the system and retrieve the address when it is needed. The physical address, often referred to as the Media Access Control (MAC) address in a networking system, uniquely identifies each node of a network. For example, each Ethernet card has a unique MAC address that identifies it on the network. The Ethernet MAC address is 48 bits (6 bytes) in length. The 48-bit serial number in the DS2401 device can be utilized as the MAC address. This 1-Wire device interface retrieves the ROM number in the 1-Wire device by eight sequential byte transfers on the data port, verifies the data by calculating CRC and matching it with the retrieved CRC code from the device.

The following section of this application note is a tutorial review of DS2401 1-Wire device. The next section describes the 1-Wire device interface in detail, and the last section summarizes the results.

DS2401 Silicon Serial Number Review

Hardware Configuration

Communication with the 1-Wire device is via the 1-Wire bus, which has only a single line. The 1-Wire bus has a single bus master system and one or more slaves. In all instances, the DS2401 is a slave device, while the interface module in the FPGA is the bus master. Only one DS2401 device is attached to the bus in this application, although the 1-Wire bus has the capability to function properly while having multiple slave devices attached to it.

The bus master requires a pull-up resistor at the master end of the bus and has a 3-state output or open drain connection to the bus. The value of the pull-up resistor should be approximately 5 kΩ for short line lengths. Virtex or Spartan-II FPGAs have optional internal pull-up resistors available on each I/O pin. In order to avoid an external pull-up resistor on the 1-Wire bus, internal pull-up resistor on the FPGA I/O pin for the 1-Wire connection can be activated. This pin should be a bidirectional pin so that the FPGA can send and receive the data on the same data line. Figure 2 shows the application circuit for 1-Wire interfacing between the FPGA and the DS2401 device.

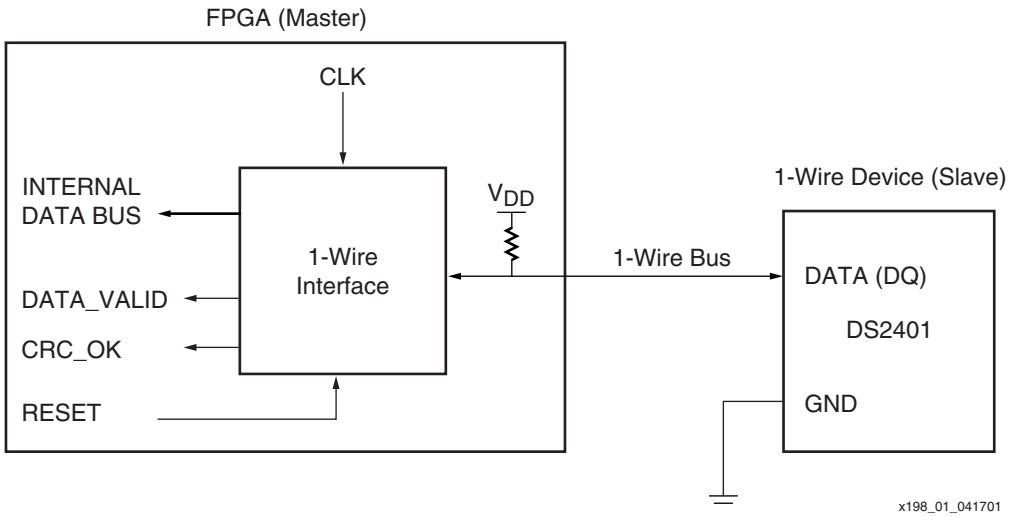

Figure 2: **1-Wire Interfacing Circuit**

Transaction Sequence

The 1-Wire bus requires strict signaling protocol to ensure data integrity. The protocol consists of four types of signaling on the bus: 1-Wire Initialization (Reset Pulse and Presence Pulse), Write “0”, Write “1”, and Read Data. All transactions on the 1-Wire bus should follow the protocol. The specific sequence for retrieving ROM number from the DS2401 via the 1-Wire interface is as follows:

1. **1-Wire Initialization**
2. **Send Read ROM Command**
3. **Read ROM Data**

1-Wire Initialization

The 1-Wire Initialization is required to begin any transaction on the 1-Wire bus, as shown in Figure 3. The 1-Wire master (FPGA) first transmits a Reset Pulse on the 1-Wire bus for T_{RSTL}. The 1-Wire bus line is then pulled High by the pull-up resistor. If a slave (e.g., DS2401) is on the bus and ready to operate, it will wait for T_{PDH} after detecting the rising edge on the bus, and then acknowledges by transmitting a Presence Pulse for T_{PDL}. The master should sample and validate the Presence Pulse during T_{PDL} in order to move to the next step.

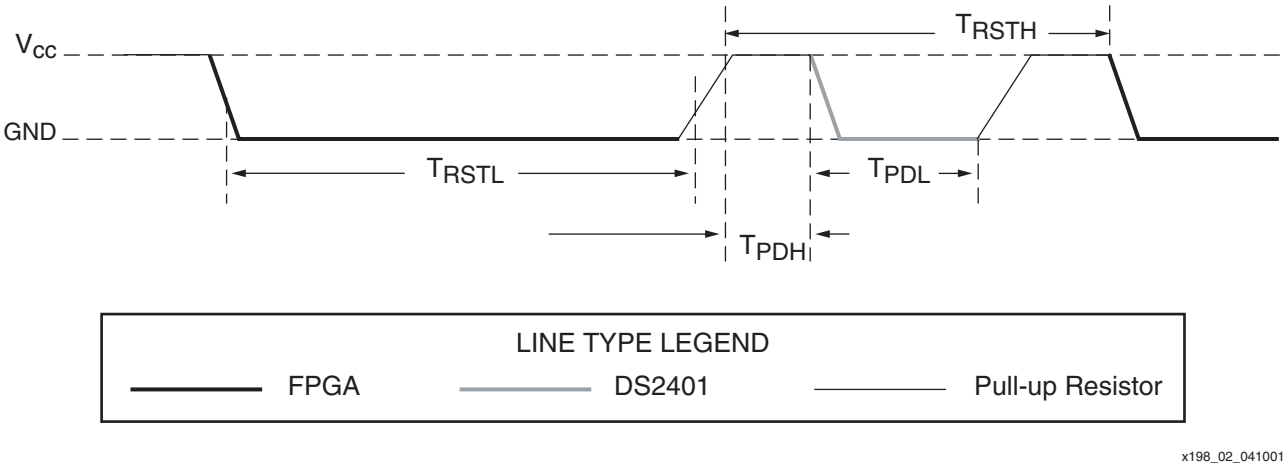

Figure 3: **1-Wire Initialization (Reset Pulse and Presence Pulse)**

Send Read ROM Command

Once the bus master has detected the presence of a 1-Wire device, it can issue a Read ROM function command by sending [33h] to the 1-Wire bus. This command allows the master to read the 64-bit ROM data from DS2401. This command can only be used if there is a single slave device on the bus, as in this application. The sequence of sending the command consists of writing 8 bits of data sequentially to the 1-Wire bus by sending the LSB first. Writing a bit of data occupies one Write Time Slot, based on the Write "0" and Write "1" 1-Wire signaling protocol, as illustrated in Figure 4. The master driving the bus Low initiates all time slots. DS2401 samples the bus line between 15 µs and 60 µs.

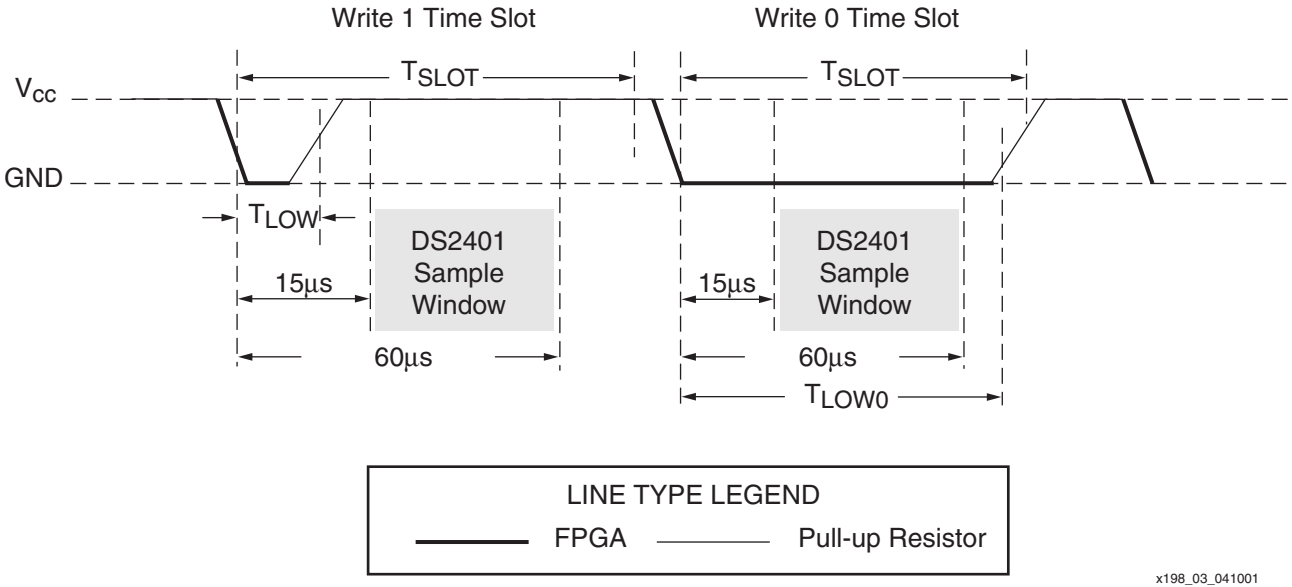

Figure 4: **1-Wire Write Time Slots (Write 1 and Write 0)**

Read ROM Data

After sending the Read ROM command, the 1-Wire master can read the ROM data from DS2401 by initiating Read Time Slots and sampling the data on the bus at the proper time. The master needs to drive the bus Low for at least 1µs and then releases it. DS2401 either holds the line Low for responding a "0", or releases it immediately for a "1". The master should sample the bus at T_{RDV} and end the Read Time Slot after T_{SLOT}, as shown in Figure 5. By sequentially repeating the Read Time Slots, the master can retrieve the 64-bit ROM data from DS2401. Data is available by LSB first and MSB last, which will be the family code (`01h` for DS2401) for the first byte, then six bytes of the serial number, and one byte of CRC code at the end.

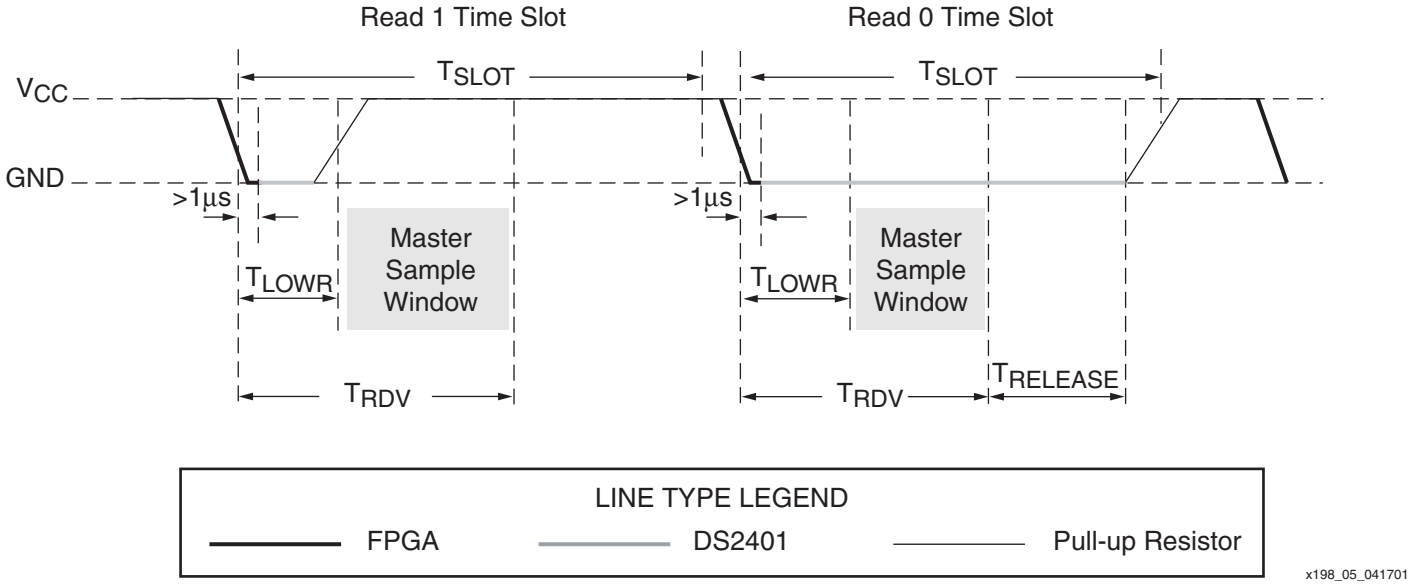

Figure 5: **1-Wire Read Time Slots (Read 1 and Read 0)**

Table 1: **Timing Specifications in 1-Wire Signaling Protocol**

Symbol	Parameter	MIN	MAX	Units
T_{RSTL}	Reset Time Low	480		µs
T_{RSTH}	Reset Time High	480		µs
T_{PDL}	Presence Detect Low	60	240	µs
T_{PDH}	Presence Detect High	15	60	µs
T_{SLOT}	Time Slot	60	120	µs
T_{LOW0}	Write "0" Low Time	60	120	µs
T_{LOW1}	Write "1" Low Time	1	15	µs
T_{LOWR}	Read Data Low Time	1	15	µs
T_{RDV}	Read Data Valid	Exactly 15		µs
$T_{RELEASE}$	Release Time	0	45	µs

CRC Generation

To validate the data retrieved from DS2401, the 1-Wire master calculates a CRC value from the received data except for the last byte. The master compares this generated value with the last byte of data from DS2401. If the two CRC values match, the retrieved data is correct. The equivalent polynomial function of the CRC is $CRC = x^8 + x^5 + x^4 + 1$, as illustrated in Figure 6.

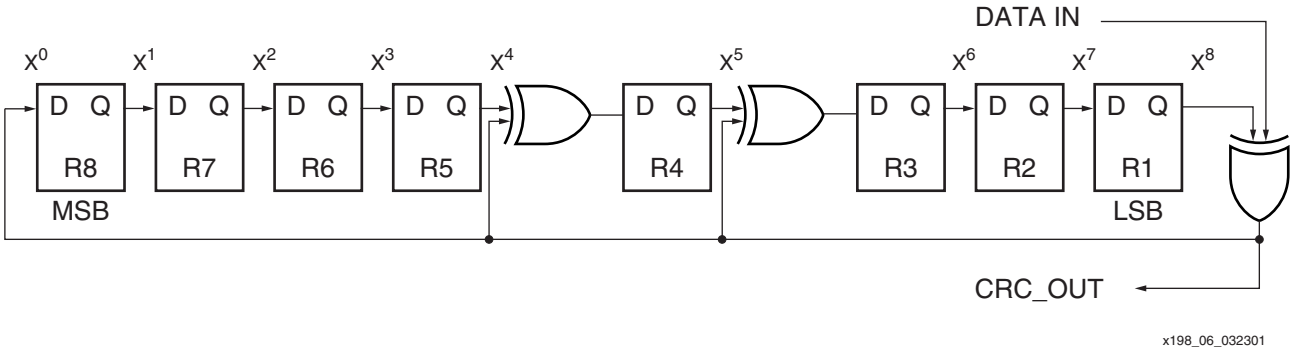

Figure 6: **DS2401 8-Bit CRC Generation**

1-Wire Interface

This interface design, available in both VHDL and Verilog, can be mapped into the user's system with very few internal interfacing signals. The design uses 55 Configurable Logic Block (CLB) slices (52 registers and 65 LUTs). It has the following features:

- Interface with the 1-Wire device using only one I/O port
- Capable of running off any system clock from 3 MHz to 80 MHz.
- The OW_MASTER module is capable of running off any internal clock from 0.6 MHz to 1.3 MHz.
- Programmable and removable CRC generator
- Uses FPGA internal pull-up component on the I/O port or an external pull-up resistor
- Retrieved ROM number from 1-Wire device is available by bytes and is indicated by a valid signal. When crc_ok is asserted, the 48-bit ROM serial number is also available on the 48-bit parallel output.

Figure 7 shows the blocks in the top-level reference design. The top-level design has two main blocks, the CLK_DIVIDER (clk_div.vhd) and the OW_MASTER module (onewire_master.vhd). The CLK_DIVIDER module is a programmable clock divider. It provides the desired frequency for the OW_MASTER module using a system clock on the sys_clk pin. By changing a generic of the CLK_DIVIDER module, the top-level design supports any system clock frequency from 3 MHz to 80 MHz. An FPGA internal pull-up component is activated on the bi-directional I/O port (dq) if the generic ADD_PULLUP is turned on. This port is the single connection with the 1-Wire device (DS2401).

Communication with the 1-Wire device starts after the sys_reset is deactivated. When data_valid signal is asserted, one byte of retrieved data is available on the 8-bit wide data bus. When all 8 bytes of data have been received error-free, the crc_ok signal is asserted as a latch signal. Meanwhile, the 48 bits ROM serial number (sn_data) is available at the parallel output. Either the 8-bit data output or 48-bit data output can be left open if it is not used by the user system. When using a typical 1 MHz internal clock for the OW_MASTER, it takes about 7 µs to finish the process of retrieving the ROM number from the 1-Wire device.

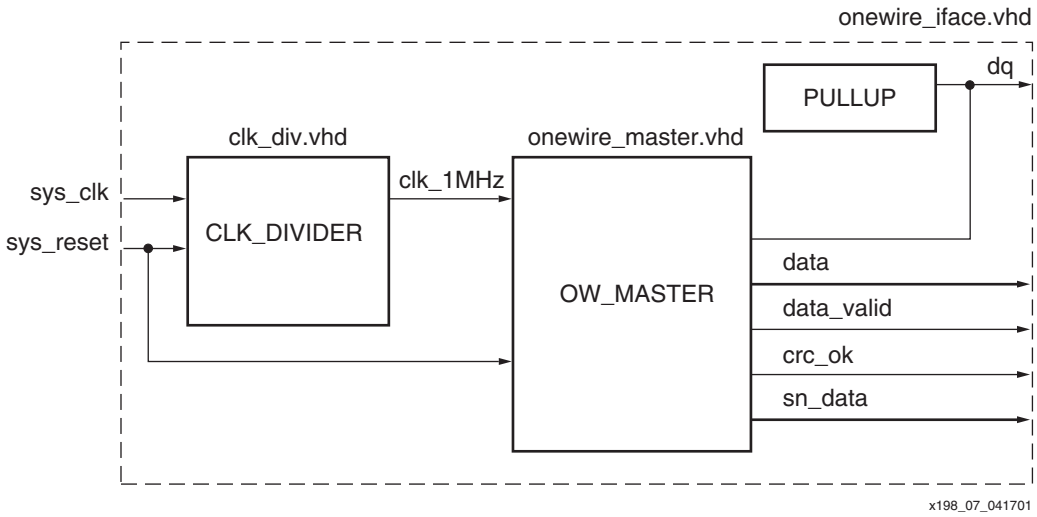

Figure 7: **Top Level Block Diagram**

The OW_MASTER module, as shown in Figure 8, contains a Finite State Machine (FSM) which provides the 1-Wire timing control and data flow control. All 1-Wire timing patterns are generated in OW_MASTER module using an internal clock (clk_1MHz). The clock frequency can range from 0.6 MHz to 1.3 MHz, but is typically 1 MHz. The module uses two Johnson counters to generate a 50-kHz slow clock and identify 1 µs and 20 µs time slots. The module uses two shift registers to identify each bit and byte in the data stream, and identify 80 µs time slots as well.

A bit register is used as the data buffer to assemble 8 bits to one data byte. A byte register is used to assemble the 6 bytes of serial number for the 48-bit parallel output. The module uses a CRC generator to calculate CRC value based on the received data, and match it with received CRC code to assert crc_ok output. The CRC generator uses 10 LUTs, 15 registers, and 16 slices. If necessary, the CRC generator can be removed by switching off a generic in the design to save FPGA resources.

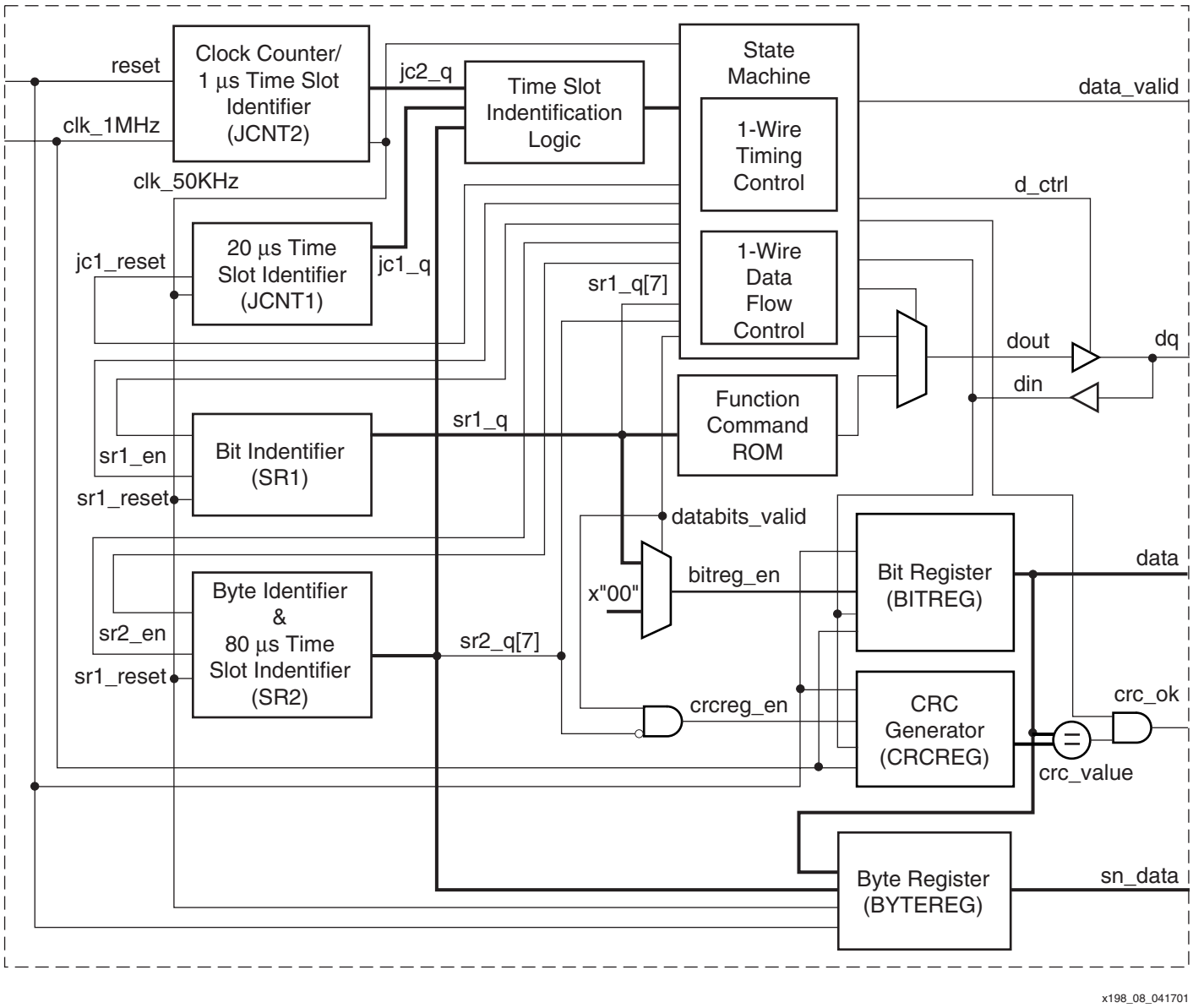

Figure 8: **Block Diagram of OW_MASTER**

Pin Descriptions

The pin descriptions are outlined in Table 2, organized according to system or 1-Wire device interface.

Table 2: **Pin Descriptions**

Interface	Pin		Width	Notes
	Name	Direction		
System	sys_clk	In	1	Supports 3 MHz to 80 MHz
	sys_reset	In	1	System reset
	data	Out	8	LSB byte first; MSB byte last
	data_valid	Out	1	20 µs pulse if internal clock for the OW_MASTER is 1 MHz
	crc_ok	Out	1	A latch signal to indicate CRC checking is OK. If CRC checking is disabled, it indicates that all bytes have been received.
	sn_data	Out	48	Parallel output of the ROM serial number. It is valid when crc_ok is active.
1-Wire Device	dq	In/Out	1	Maximum data rate of 16.3 kb/s

Clock Divider

The OW_MASTER module generates all 1-Wire timing patterns using a typical base clock of 1.0 MHz. The 1-Wire interface generates this clock frequency internally using a Clock Divider given an external reference on the sys_clk pin. The external clock must have a frequency in the range of 3 MHz to 80 MHz. The Clock Divider cascades two 16-bit Shift Register LUTs (SRL16) and one register, as shown in Figure 9, to generate the internal clock. The data (D) is loaded into the first bit of the shift register during the Low-to-High clock (CLK) transition.

During subsequent Low-to-High clock transitions, data is shifted to the next highest bit position as new data is loaded. The data appears on the Q output when the shift register length determined by the address inputs is reached. The Clock Divider uses a generic (CLK_DIV) to specify the length of the shift register to determine the period of the output clock. Table 3 shows the proper value of the generic based on the external reference frequency. For example, the user can write `0Bh` or `0Ch` to the generic when providing a 50-MHz input clock. Thus, the frequency of the output clock is 0.9 MHz or 1.04 MHz.

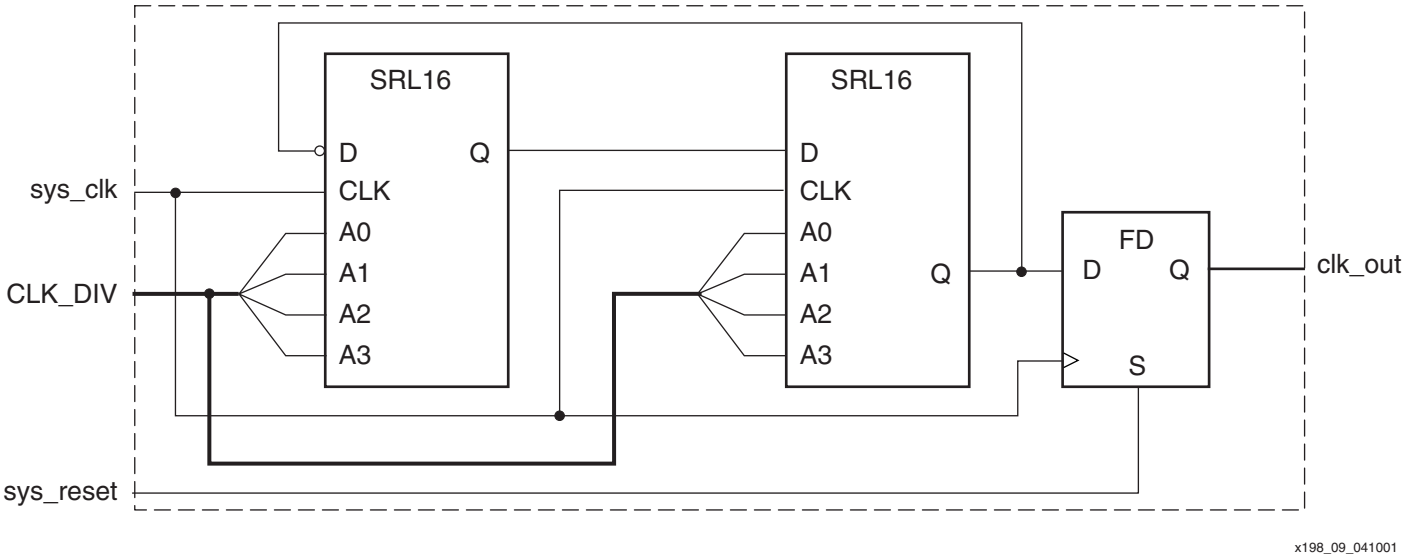

Figure 9: **Clock Divider**

Table 3: **Generic Settings in Clock Divider Based on Input Clock Rates**

Min. Input Clock Frequency (MHz)	Max Input Clock Frequency (MHz)	Divider Ratio	CLK_DIV Value
3	5	4	0
5	9	8	1
9	14	12	2
14	18	16	3
18	22	20	4
22	26	24	5
26	30	28	6
30	34	32	7
34	38	36	8
38	42	40	9
42	46	44	10
46	50	48	11
50	54	52	12
54	58	56	13
58	62	60	14
62	80	64	15

State Machine

Figure 10 shows the overview of the FSM in the master (short for the OW_MASTER module).

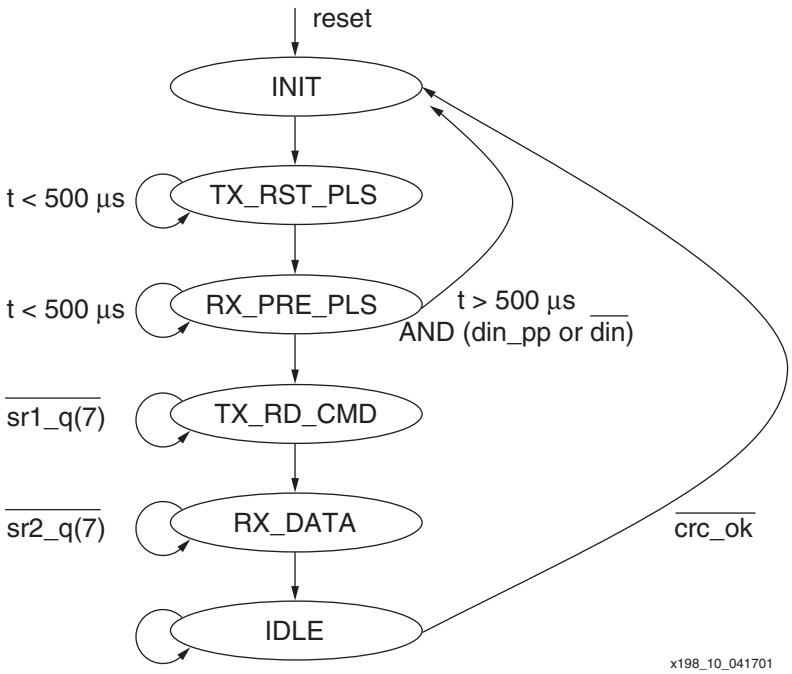

Figure 10: **State Machine Overview**

Assume a typical 1-MHz clock drives the master. Thus, the master generates an internal slow clock (50 kHz in this case) to synchronize the FSM. The FSM integrates the functionality of the timing and data flow control on the 1-Wire bus. Based on the 1-Wire signaling protocol, the timing control uses discrete Time Slots to differentiate time periods in the whole process. The shortest time slot is 1 µs long. State transitions usually occur at a specific time point and/or when a specific data transaction completes. To determine any time point or time slot, the master uses combinational logic of the outputs from the Johnson Counters (JCNT1 and JCNT2) and the Shift Register (SR2).

When getting a system reset, the FSM is initially in the INIT state. It goes into the TX_RST_PLS state automatically at the next clock cycle.

During the TX_RST_PLS state, the master issues a reset pulse on the 1-Wire bus for 480 µs. To determine such long timing periods, the FSM enables the SR2 to shift every 80 µs for six times. The 80 µs Time Slot is generated by JCNT1 by counting a 20-µs clock period for four times. Assume that entering this state is at time “0”. The FSM releases the 1-Wire bus and enables a state transition at 480 µs, and moves into the RX_PRE_PLS state at next clock cycle (at 500 µs). Because of the presence of the pull-up resistor on the 1-Wire bus, it takes less than 2 µs for the bus to go back High.

During the RX_PRE_PLS state, the master waits and detects the Presence Pulse from the 1-Wire device. The occurrence of the Presence Pulse is predicable in a time window of 15 µs to 60 µs after the 1-Wire bus goes back High. The Pulse will remain for at least 60 µs. Suppose entering this state is at time “0”. It is safe to register the data (din_pp) on the 1-Wire bus at 80 µs. The FSM should wait for enough time before the slave device releases the bus and the 1-Wire bus settles. The FSM enables a state transition at 480 µs. If din_pp is Low (Presence Pulse detected), and din is High (pullup on the 1-Wire bus detected), then the FSM will move to the next state at 500 µs, otherwise it goes back to the INIT state to perform a retry.

During the TX_RD_CMD state the master sends the 8-bit Read ROM command to the 1-Wire device. The master writes “0” or “1” to the 1-Wire bus for each bit in the command. Writing of each bit occupies one Write Time Slot of 80 µs. Assume that the beginning of each Time Slot

is at time “0”. Each Time Slot begins with the master pulling down the 1-Wire bus for 10 µs. Data writing is performed from 10 µs to 60 µs. If writing “0”, the master keeps pulling down the 1-Wire bus for 50 µs, then releases the bus. If writing “1”, the master immediately releases the bus. The bus goes back High in less than 2 µs. The master repeats this process for eight times in order to write all the 8 bits of data. A Shift Register (SR1) is used to count for the 8 bits by shifting each bit at the end of every Write Time Slot. When the master finishes writing all the bits (`sr1_q(7) = “1”`), it moves to next state.

During the RD_DATA state, the master starts to receive data coming from the 1-Wire device. The master utilizes both shift registers in this state. SR1 is used to count for the 8 bits in each byte, while SR2 is used to count for total 8 bytes of data. Similar to the Writing, the master uses Read Time Slot of 80 µs for each data bit. Suppose that the beginning of each Time Slot is at time 0. Each Slot begins with the master pulling down the 1-Wire bus for 1 µs. Then the master releases the bus immediately to allow the slave device to occupy the bus.

The master samples the data on the bus from 14 µs to 15 µs. Meanwhile, the FSM generates the enable signals for the BITREG and CRCREG. The master feeds the data into the Bit Register (BITREG) and CRC Generator (CRCREG); therefore, the data is buffered and the CRC is calculated at the same time. The master repeats the same process for 8 times in order to retrieve one byte of data, then it asserts the data_valid signal to indicate availability of a valid data byte on the data bus. After retrieving 8 bytes of data (`sr2_q(7) = "1"`), the FSM moves to next state.

The last state is the IDLE state. The master enables and latches the crc_ok output. If CRC checking is enabled and CRC error is detected, the FSM will go back to the INIT state. Otherwise, the FSM stays in the IDLE state if no further system reset occurs.

Timing Diagrams

The timing diagrams show the behavior of the signals in OW_MASTER and show the typical data patterns on the 1-Wire bus. The timing diagrams are based on 1 MHz internal clock of the OW_MASTER module.

1-Wire Initialization

Figure 11 is the timing diagram for the 1-Wire initialization, including INIT, TX_RST_PLS, and RX_PRE_PLS states. At T_1, the 1-Wire interface (master) issues a Reset Pulse to the 1-Wire bus for 480 µs. Within T_8 to T_9, the 1-Wire device (slave) responses a Presence Pulse. The master samples the 1-Wire bus at T_9 into the register din_pp.

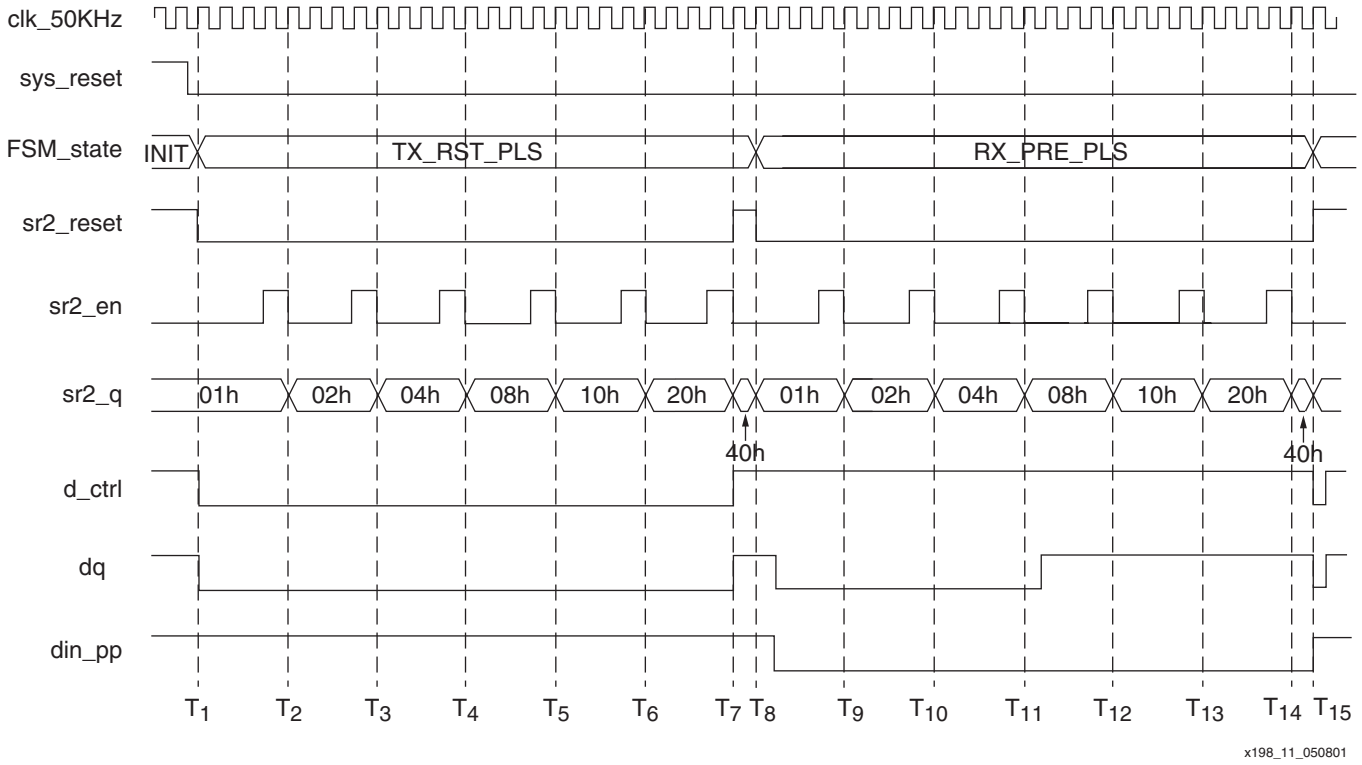

Figure 11: **1-Wire Initialization Timing Diagram**

Write Read ROM Command

Figure 12 is the timing diagram for writing the Read ROM function command (`33h`) to the 1-Wire bus (TX_RD_COMMAND state).

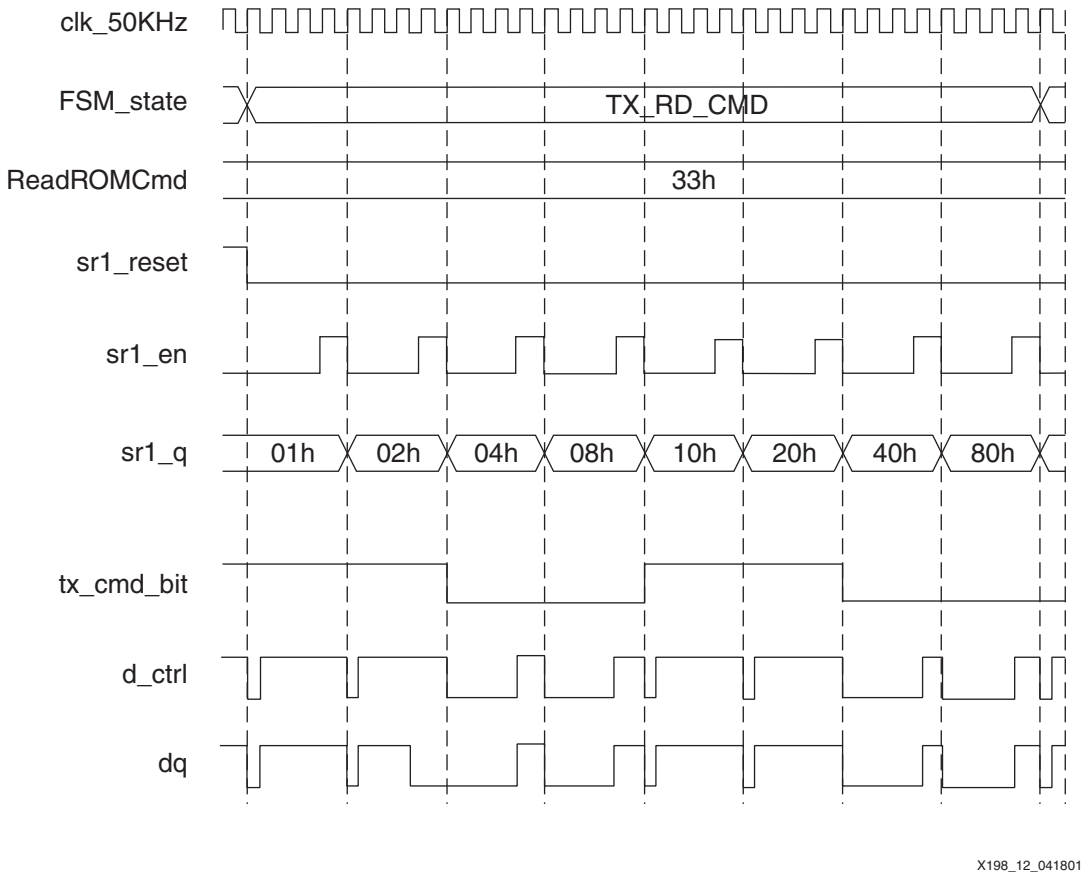

Figure 12: **Write Timing Diagram**

Read a Byte Cycle

Figure 13 is the timing diagram for reading eight data bits from the 1-Wire device (RD_DATA state). The master samples the 1-Wire bus during 14 µs to 15 µs for each bit of data, which is indicated by the databit_valid signal. The data is received with LSB first. This timing diagram shows the first 8 bits of data received from the DS2401 1-Wire device. The data is the family code (`01h`) of the DS2401 device. The DS2430A will respond with a family code of `14h` instead of `01h`.

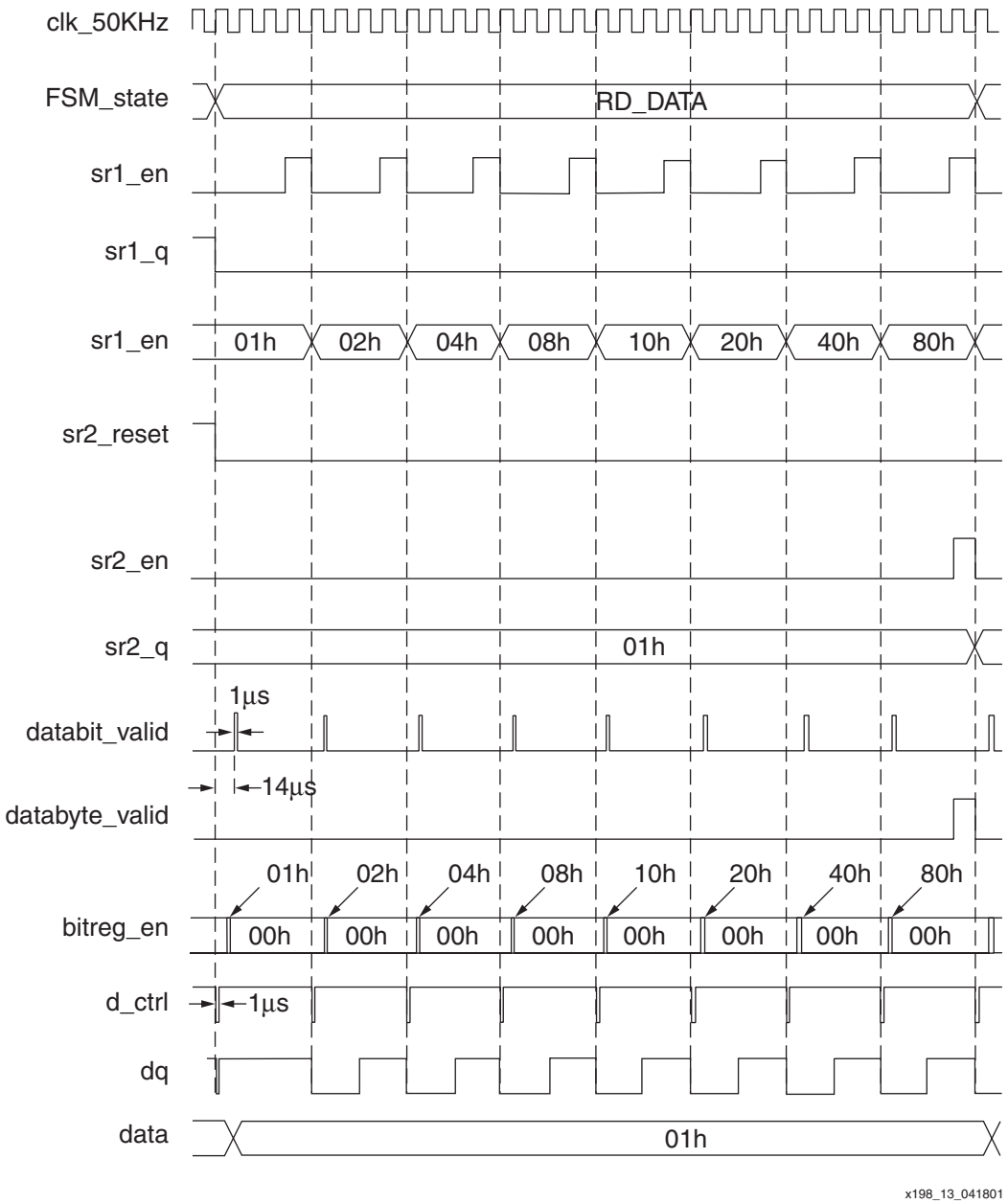

Figure 13: **Read Timing Diagram**

Summary

This application note and reference design describe an interface between a Virtex or Spartan-II series FPGA and 1-Wire device to retrieve the ROM number from the 1-Wire device. The design is developed using HDL, and it can be easily modified for a different system with a system clock range from 3 MHz to 80 MHz. Due to its small size, it can be easily mapped into user system with simple internal interface and a minimal external interface to the 1-Wire device. The reference design can be downloaded from the Xilinx web site at:

ftp://ftp.xilinx.com/pub/applications/xapp/xapp198.zip

References

1. Dallas Semiconductor, DS2401 Silicon Serial Number, Data Sheet, 1999.
2. Dallas Semiconductor, DS2430A 256-Bit 1-Wire EEPROM, Data Sheet, 1999.
3. Dallas Semiconductor, Book of Button Standards, Data Book, 1997.
4. Dallas Semiconductor, DS1WM Synthesizable 1-Wire Bus Master, Application Note, 1999.
5. Dallas Semiconductor, Communicating Through the 1-Wire Master, Application Note, 2000.
6. Dallas Semiconductor, Understanding and Using Cyclic Redundancy Checks with Dallas Semiconductor iButton™ Products, Application Note, 1999.
7. Xilinx Inc., Virtex 2.5V Field Programmable Gate Arrays, Data Sheet, 2000.
8. Xilinx Inc., Spartan-II 2.5V., Field Programmable Gate Array Data Sheet, 2000.

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
05/08/01	1.0	Initial Xilinx release.