Please note that Cypress is an Infineon Technologies Company.
The document following this cover page is marked as “Cypress” document as this is the company that originally developed the product. Please note that Infineon will continue to offer the product to new and existing customers as part of the Infineon product portfolio.

Continuity of document content
The fact that Infineon offers the following product as part of the Infineon product portfolio does not lead to any changes to this document. Future revisions will occur when appropriate, and any changes will be set out on the document history page.

Continuity of ordering part numbers
Infineon continues to support existing part numbers. Please continue to use the ordering part numbers listed in the datasheet for ordering.

CY8C20045, CY8C20055, CY8C20xx6, CY8C20xx6A, CY8C20xx6AS, CY8C20xx6AN, CY8C20xx6L, CY8C20xx6H, CY7C643xx, CY7C604xx, CY8CTST2xx, CY8CTMG2xx, CY8C20xx7, CY8C20xx7S, and CY8C20xx7AN

ISSP Programming Specifications

Document #: 001-57631 Rev. *I

August 30, 2022

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

Phone (USA): 800.858.1810
Phone (Intnl): 408.943.2600
http://www.cypress.com

© Cypress Semiconductor Corporation, 2009-2022. This document is the property of Cypress Semiconductor Corporation, an Infineon Technologies company, and its affiliates (“Cypress”). This document, including any software or firmware included or referenced in this document (“Software”), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress’s patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. No computing device can be absolutely secure. Therefore, despite security measures implemented in Cypress hardware or software products, Cypress shall have no liability arising out of any security breach, such as unauthorized access to or use of a Cypress product. CYPRESS DOES NOT REPRESENT, WARRANT, OR GUARANTEE THAT CYPRESS PRODUCTS, OR SYSTEMS CREATED USING CYPRESS PRODUCTS, WILL BE FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (collectively, “Security Breach”). Cypress disclaims any liability relating to any Security Breach, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from any Security Breach. In addition, the products described in these materials may contain design defects or errors known as errata which may cause the product to deviate from published specifications. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. “High-Risk Device” means any device or system whose failure could cause personal injury, death, or property damage. Examples of High-Risk Devices are weapons, nuclear installations, surgical implants, and other medical devices. “Critical Component” means any component of a High-Risk Device whose failure to perform can be reasonably expected to cause, directly or indirectly, the failure of the High-Risk Device, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from any use of a Cypress product as a Critical Component in a High-Risk Device. You shall indemnify and hold Cypress, including its affiliates, and its directors, officers, employees, agents, distributors, and assigns harmless from and against all claims, costs, damages, and expenses, arising out of any claim, including claims for product liability, personal injury or death, or property damage arising from any use of a Cypress product as a Critical Component in a High-Risk Device. Cypress products are not intended or authorized for use as a Critical Component in any High-Risk Device except to the limited extent that (i) Cypress’s published data sheet for the product explicitly states Cypress has qualified the product for use in a specific High-Risk Device, or (ii) Cypress has given you advance written authorization to use the product as a Critical Component in the specific High-Risk Device and you have signed a separate indemnification agreement.

Cypress, the Cypress logo, and combinations thereof, PSoC, CapSense, EZ-USB, F-RAM, Traveo, WICED, and ModusToolbox are trademarks or registered trademarks of Cypress or a subsidiary of Cypress in the United States or in other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.

Contents

1. Overview **5**

1.1 Introduction 5
1.2 Host Programmer - CY8C20xx6A Programming Interface 6
1.2.1 Programming Pin Drive Modes....6
Document Revision History 7

2. Programming Flow **9**

2.1 Target Programming 9
2.1.1 Vectors....9
2.1.2 Clocking....10
2.1.3 Command Format....10
2.1.4 Wait-and-Poll 10
2.1.5 Programming Pin Drive Modes....11
2.2 Insertion Check 11
2.3 Verify Silicon ID....12
2.3.1 Reset Mode 13
2.3.2 Power-on Mode 13
2.3.3 Power-on Mode with External Supply....14
2.3.4 Read-ID-Word 14
2.4 Rerun Check (Optional) 14
2.5 Erase....15
2.6 Program and Verify 15
2.7 Set Security....15
2.8 Verify Security (Optional) 16
2.9 Device Checksum 17
2.10 Power Down....17
2.11 Verify (Optional) 17
2.12 Specifications and Definitions 18
2.12.1 DC Programming Specifications....18
2.12.2 AC Programming Specifications 18
2.12.3 Device Address and Block Definitions 19

A. Programming Mnemonics for CY8C20xx6, CY8C20xx7, CY7C643xx, CY7C604xx **23**

A.1 Bit Streams for Mnemonics....23

1. Overview

1.1 Introduction

In-circuit programming is convenient for prototyping, manufacturing, and in-system field updates. CY8C20045, CY8C20055, CY8C20xx6A, and CY8C20xx7 devices can be programmed using the in-system serial programming (ISSP) protocol, a proprietary protocol used by Cypress.

This reference manual provides the information developers and programmer vendors need to create their own in-system programming solutions for CY8C20045, CY8C20055, CY8C20xx6A, and CY8C20xx7 devices. The following topics are covered in this document:

- Information on how to interface a host programmer with CY8C20045, CY8C20055, CY8C20xx6A, and CY8C20xx7 devices
- Description of the ISSP protocol
- AC/DC programming specifications
- Programming vectors
- Introduction to the Intel hex file format

The programming procedure involves a programmer and a target device. The programmer communicates serially with the target, supplies the clocking, and sends commands to the target. The target receives data from the programmer and supplies data upon a read request. It only drives the data line when it receives a request from the programmer. The programmer programs the target with the program image contained in the *<PROJECT NAME>.hex* file, which is generated by PSoC Designer™.

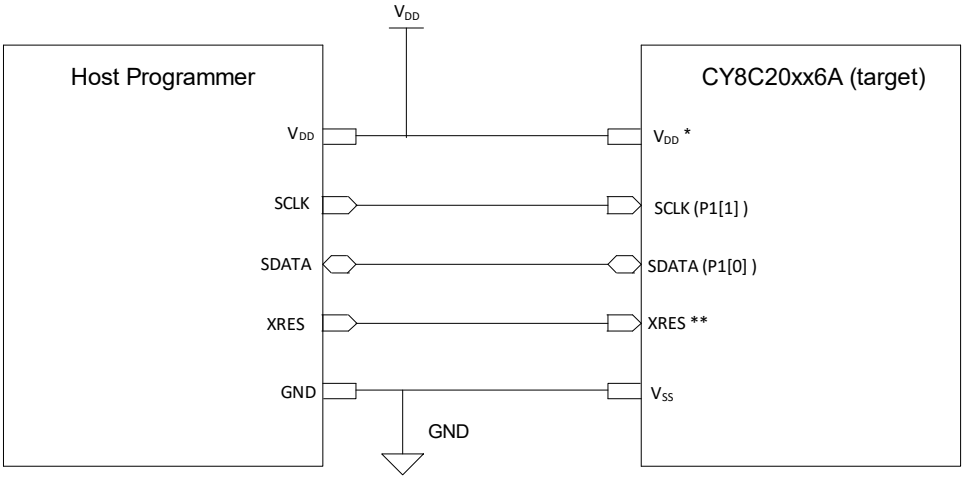
Keep the following things in mind when you are developing a host programming application:

- The programming vectors provided in this document should not be compared with those generated by the MiniProg1, MiniProg3, or ICE-Cube. MiniProg1, MiniProg3, and ICE-Cube use a different version of the protocol to program the target device. Cypress recommends using the programming vectors provided in this document to develop your host side interface and program CY8C20045, CY8C20055, CY8C20xx6A, and CY8C20xx7 devices.
- Even though the ISSP protocol uses a bidirectional data line for communication between a host and a target device, it is not related to the I^2C protocol.

1.2 Host Programmer - CY8C20xx6A Programming Interface

Figure 1-1 shows the connections between the host programmer and the target CY8C20xx6A device. If you are using the MiniProg1 programmer, refer to the Knowledge Base article, *Part Number for MiniProg1 Target Connector, 5-pin ISSP Header in MiniProg3*, for details.

Figure 1-1. Host Programmer - CY8C20xx6A Interface



* To program in Power Cycle mode, the host programmer must be capable of toggling power to the CY8C20xx6A device.

** XRES pin in CY8C20xx6A is active high input. It has an internal pull -down resistor to keep it at logic low when left floating. XRES pin is not available in all device packages. Check the device data sheet for information on XRES pin availability. Use Power Cycle mode if XRES is not available.

1.2.1 Programming Pin Drive Modes

The electrical pin connections between the programmer and the target device shown in Figure 1-1 are listed in Table 1-1. This includes two signal pins, a reset pin, a power pin, and a ground pin. Leave the other pins floating. The pin naming conventions and drive strength requirements are also listed in Table 1-1.

Table 1-1. Pin Names and Drive Strengths

Pin Name	Function	Programmer HW Pin Requirements	CY8C20xx6A Drive mode behavior
P1[0]	SDATA – Serial data in/out	Drive TTL Levels, Read TTL, High-Z	Strong drive (while sending data to the host), Resistive pull-down mode (reading data from the host, waiting for data from the host)
P1[1]	SCLK – Serial clock	Drive TTL Levels	High-Z digital input
XRES	Reset	Drive TTL Levels. Active High	Active high reset input with internal resistive pull down
V_{SS}	Power supply ground connection	Low Resistance Ground Connection	Ground connection
V_{DD}	Positive power supply voltage	0 V, 1.8 V, 3.3 V, 5 V. 20 mA Current Capability	Supply voltage

The CY8C20xx6A SDATA pin drive modes vary during programming. When CY8C20xx6A drives the SDATA line to indicate that it has started up completely or to send data back to the host, it is in a strong drive configuration. When CY8C20xx6A waits for data or receives data from the host, SDATA is in a resistive pull-down configuration. It is important to design the host external pin drive mode circuitry to detect a strong high to resistive low transition, and to be able to drive the SDATA pin both high and low when it is in resistive pull down mode. Because the SDATA line has a internal pull-down resistor (5.6 kΩ), external pull-up resistors can cause the host to miss high-to-low transitions on the target device due to resistive voltage divider. Therefore, using external pull-up resistors on the SDATA line is not recommended.

Document Revision History

Document Title: CY8C20045, CY8C20055, CY8C20xx6, CY8C20xx6A, CY8C20xx6AS, CY8C20xx6AN, CY8C20xx6L, CY8C20xx6H, CY7C643xx, CY7C604xx, CY8CTST2xx, CY8CTMG2xx, CY8C20xx7, CY8C20xx7S, and CY8C20xx7AN ISSP Programming Specifications

Document Number: 001-57631

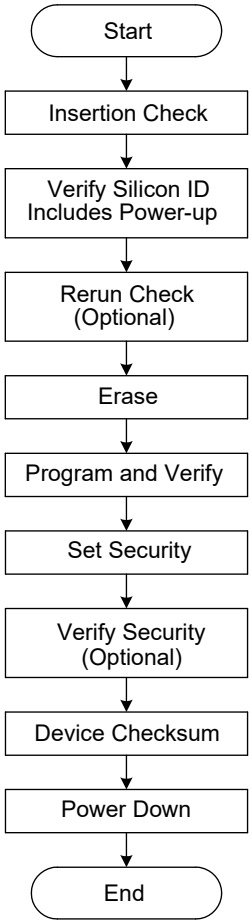
Revision	Issue Date	Origin of Change	Description of Change
**	11/18/2009	VZD	Initial revision
*A	02/09/2010	DST	1. Added AN2026c to title 2. Updated WAIT-AND-POLL diagram (Figure 11) and changed description. changed to match Programming Spec. 3. Added SYNC-ENABLE to Program and Verify Procedure Flowchart (Figure 5) to match Programming Spec. 4. Updated DC Specs table to match Programming Spec. 5. Added support for TMG2xx and TST2xx families. 6. Removed values from Figure 13and Figure 14
*B	11/25/2010	KPOL	Added support to CY7C643xx and CY7C604xx devices: Updated table-4 device address and block definition. Updated table-5 programming vectors. Added support to CY8C20746A,CY8C20766A: Updated table-4 device address and block definition. Updated table-5 programming vectors. Updated min timing value of TXRES and TVDDXRES in AC Programming Specifications. Updated WAIT-AND-POLL Sequence Timing. Added note providing reference to the knowledge base article in Appendix A.
*C	12/15/2010	KPOL	Post to external web
*D	03/24/2011	KPOL	Updated read-id word vectors; read-byte vector picture and description
*E	08/31/2011	DALE	Added programming vectors for PSoC™ Automotive Multitouch parts
*F	11/23/2011	KPOL	Converted to TRM category. Removed reference to CY8C20xx6, CY7C543xx, and CY7C604xx parts. Content updates throughout the document.
*G	03/20/2012	VMAD	Replaced Appendix A with content of Appendix 2 in Spec #001-15870 Updated title In Section 2.1.4, replaced "must apply a bit stream of 40 zero bits" with "must apply a bit stream of 30 zero bits“ Replaced Figure 2-4 with Figure 1 of Spec #001-15870 Matched data in Table 2-1 with table in 8.3.2 of Spec #001-15870 Matched data in Table 2-2 with table in 8.3.3 of Spec #001-15870 Replaced Table 2-3 with Table 7 of Spec #001-15870
*H	12/18/2012	ZINE	Added references to CY8C20045 and CY8C20055
*I	08/30/2022	PRIA	Updated to PSoC™ Automotive Multitouch guidelines

2. Programming Flow

2.1 Target Programming

For successful target programming, follow the programming flow in Figure 2-1. Each procedure is explained in detail in the following sections. Failure to complete these steps can result in incorrectly programmed flash.

Figure 2-1. Target Programming Flow



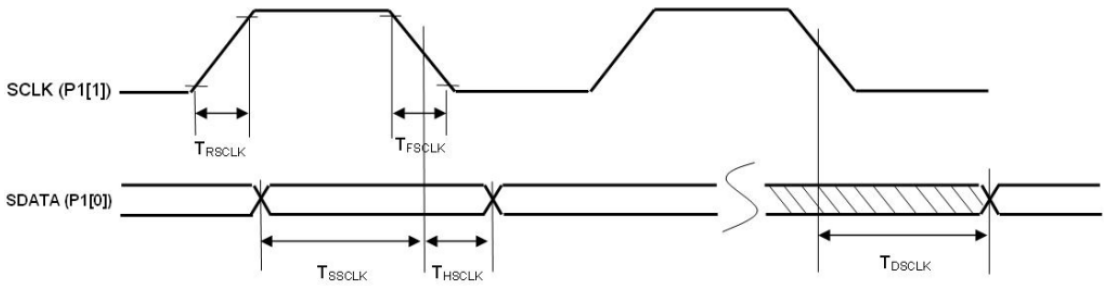
2.1.1 Vectors

Vectors are the binary representation of the commands necessary to perform various operations involved in the programming flow. Each procedure in the programming flow has many individual vectors associated with it; see Bit Streams for Mnemonics. Each vector is 22 bits long and any number of zeros can be sent between sequential vectors. The target ignores the zero padding and any subsequent '0' on the SDATA line. This continues until the target receives a '1', which is the first bit in the next vector in the vector-set.

2.1.2 Clocking

The host programmer always writes and reads on the rising edge of SCLK, while the target writes and reads on the falling edge. Figure 2-2 shows the timing waveforms of the SCLK lines. Table 2-2 provides the timing specifications mentioned in Figure 2-2. During the programming flow, the programmer supplies a clock on the SCLK to transfer data. This data transfer mode is used while the programmer communicates with the target, either by sending or receiving data. During this time, the programmer can drive the SCLK signal at any frequency that enables reliable data transfer with a maximum transmit frequency of 8 MHz (see F_{SCLK} in Table 2-2). The frequency of SCLK does not need to be accurate or consistent, as long as it is less than the 8 MHz limit. Clocks are not allowed during the Wait-and-Poll steps described in the procedure flowcharts.

Figure 2-2. SCLK Timing Diagram



2.1.3 Command Format

During programming, only the programmer drives the SCLK line. The programmer and the target can drive the SDATA line, although the target drives SDATA only upon a read request from the programmer. The programmer always writes and reads SDATA on the rising edge of SCLK, while the target writes and reads on the falling edge. After the programmer requests a read from the target, it releases the SDATA line to a high-Z state. It resumes driving the SDATA line only after the byte is sent by the target. The programmer supplies clocks even when it has released (high-Z) the SDATA line.

During the Wait-and-Poll procedure, the programmer releases (high-Z) the SDATA line and waits for a high-to-low transition on SDATA. Clocks are not allowed during the Wait-and-Poll phase. Figure 2-3 shows the Read Byte Vector Waveform.

Note The programmer must change SDATA on the positive edge of SCLK to allow enough time for the target to clock in data on the negative edge of SCLK. The programmer must show resistive drive to ground and strong drive to VDD during reads.

The programmer clocks two high-Zs between the address and data bits. When the target drives the SDATA line, the device changes SDATA on the negative edge of SCLK. The data must be read on SDATA greater than T_{DSCLK} from the falling edge (see AC Programming Specifications).

2.1.4 Wait-and-Poll

After a mnemonic bit stream is sent, the SDATA line takes 1 µs to drift low (SDATA line drifts low to V_{ILP} by the device's internal pull-down resistor). Clocking is needed before SDATA transitions from low to high. The target device pulls SDATA high while executing the mnemonic it received from the host programmer. A minimum delay of 416 ns is needed for the execution to begin. The device outputs logic high on the SDATA pin while the mnemonic is executing and switches to a logic low when the mnemonic finishes. The programmer must wait and poll the SDATA pin for the high to low transition. The maximum SDATA high time is 200 ms; see T_{POLL} from Table 2-2.

When the transition to low is observed on the SDATA line, the programmer must apply a bit stream of 30 zero bits to the SDATA pin of the device and then continue to the next mnemonic. This is shown in Figure 2-4.

Note The device drives SDATA high when the mnemonic executes and low when it is finished.

Figure 2-3. Read Byte (D7...D0) from Target CY8C20xx6A (at address A7...A0)

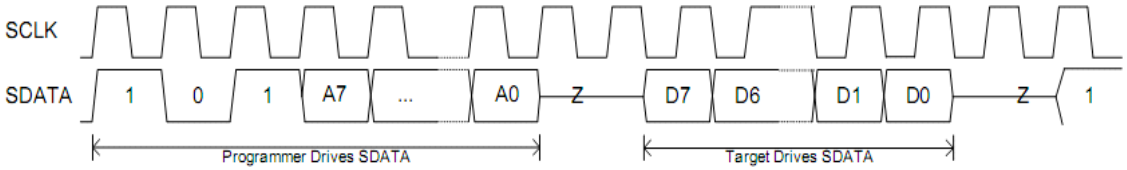
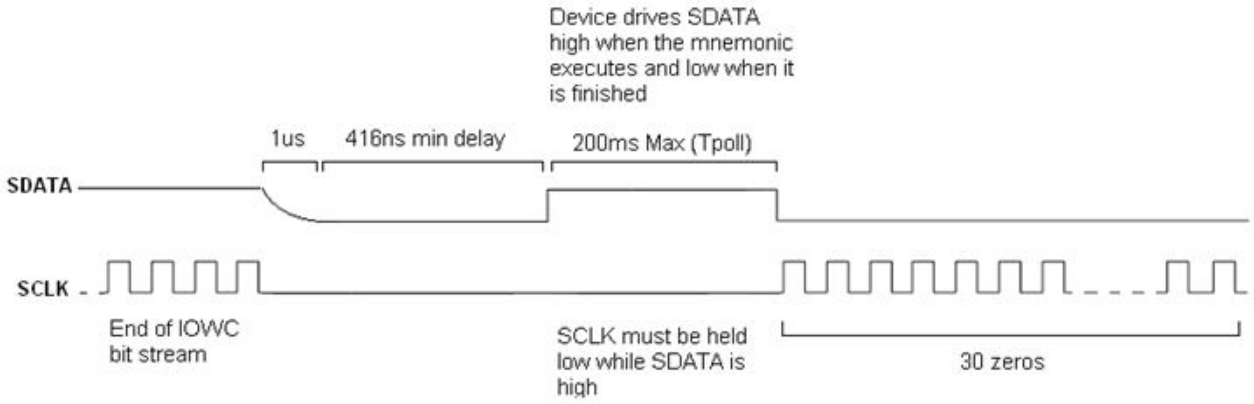


Figure 2-4. Wait-and-Poll Timing Diagram



2.1.5 Programming Pin Drive Modes

The pin drive modes vary during programming operation. When CY8C20xx6A drives the SDATA line to indicate that it has started up completely or to send data back to the host, SDATA is in a strong drive configuration. When it waits for or receives data, SDATA is in a pull-down configuration. Design the external pin drive mode circuitry so that a strong high to resistive low transition can be detected and the pin is driven both high and low when it is in pull-down mode.

2.2 Insertion Check

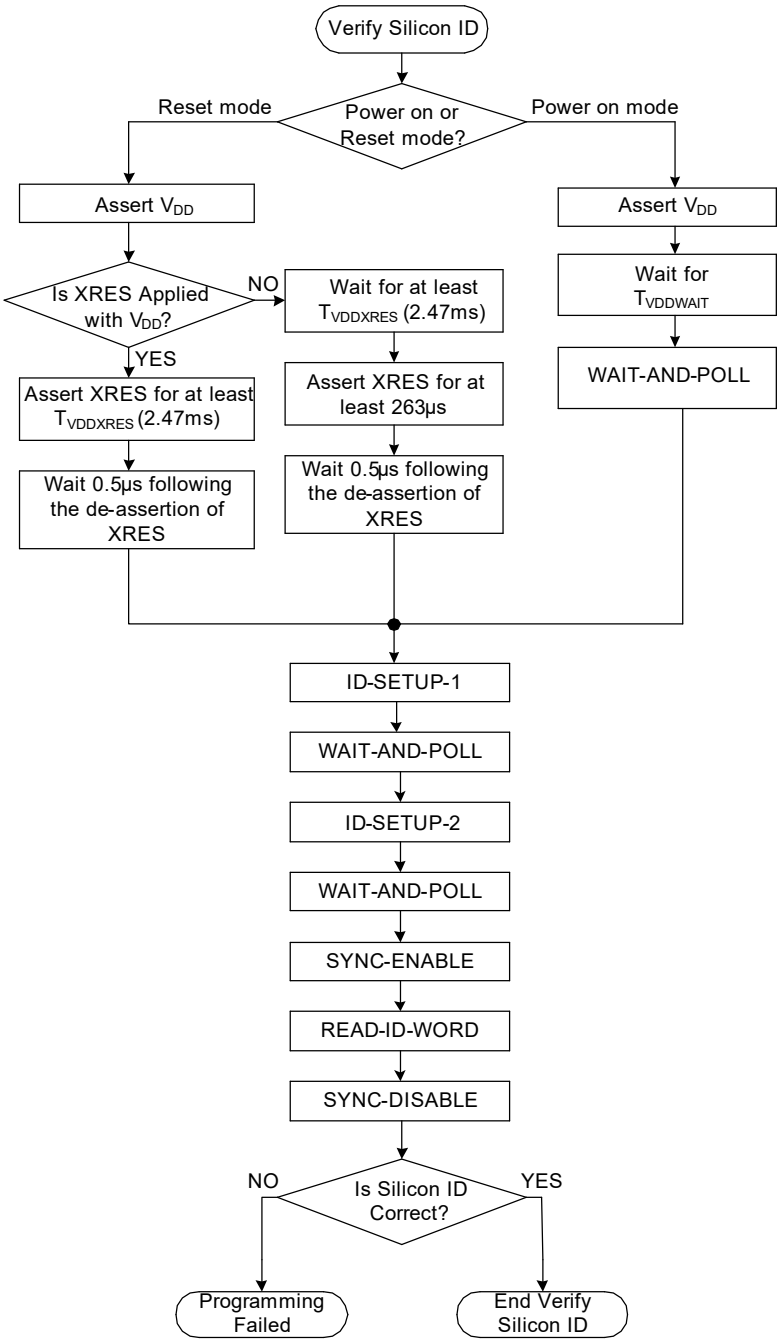
The programmer should test the physical orientation of the device before applying V_{DD}. The insertion check ensures that V_{DD} and GND of the device are correctly oriented in the socket or on the board. The programmer should execute a pin continuity test to verify electrical connection to all the required port pins listed in Table 1-1. The port pins have an industry standard reverse bias diode that can be used for the continuity test.

2.3 Verify Silicon ID

The Verify Silicon ID procedure places the chip in programming mode and reads the silicon ID of that chip. If the silicon ID does not match the expected value, the programmer must abort the programming process and send an error message to the device programmer's operator. The Verify Silicon ID procedure must be the first procedure in the flow after the insertion check and cannot be bypassed.

The programming mode is entered by the reset mode or power-on mode. If there is no XRES pin on the device, use the power-on mode. Note that because the power-on mode involves cycling power to the target, in-circuit field programming may require PCB layout considerations. If the XRES method is used, make sure that any external circuitry connected to XRES on the target board does not interfere with timing.

Figure 2-5. Verify Silicon ID Procedure



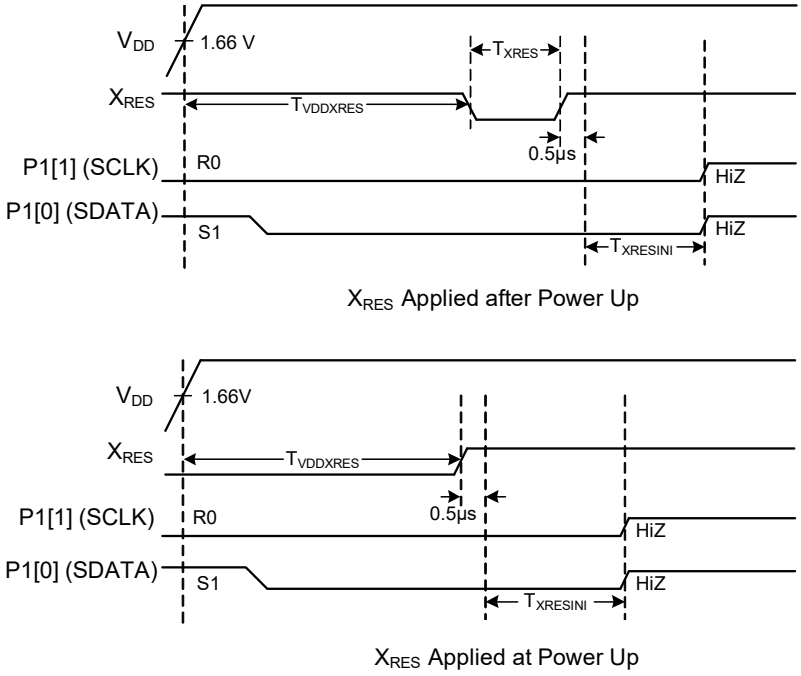
2.3.1 Reset Mode

Figure 2-6 shows the timing to enter programming mode with reset. To initialize the part using the XRES line, wait until V_{DD} is stable, then wait for $T_{VDDXRES}$ time and assert the XRES line for the time specified by T_{XRES} (see Table 2-2). The XRES line may also be brought up with the power supply line, in which case, XRES must be held high for at least $T_{VDDXRES}$. After XRES is driven low, there is a window of time specified by $T_{XRESACQ}$, (see Table 2-2), in which the first nine bits of the ID-SETUP-1 vector-set must be transmitted.

While the target executes the ID-SETUP-1 mnemonic, it drives the SDATA line high. The programmer must wait and poll the SDATA line for a high-to-low transition, which is the signal from the target that the ID-SETUP-1 mnemonic is complete.

After the ID-SETUP-1 mnemonic, send the ID-SETUP-2 mnemonic, and then wait and poll. Next, send the SYNC-ENABLE, READ-ID-WORD, and SYNC-DISABLE mnemonics. See Bit Streams for Mnemonics for mnemonic bit streams.

Figure 2-6. Acquire Sequence Timing using XRES External Reset Acquire



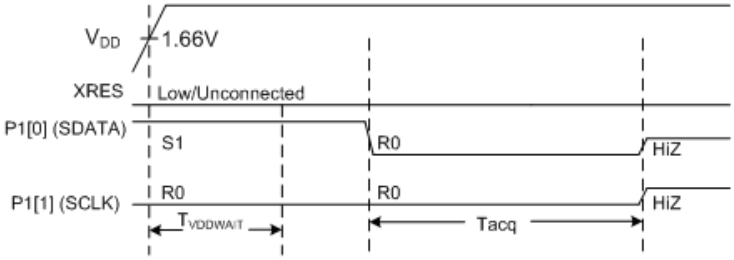
Notes

- R0 = Resistive logic ‘0’ output
- HIZ = High-impedance with logic input disabled
- $T_{VDDXRES}$ and T_{XRES} are the minimum amount of time required. Additional delay may be used without penalty
- There is a minimum required delay of 0.5 µs between the deassertion of the XRES and the entry of the first vector
- S1 = Strong logic ‘1’ output from the target

2.3.2 Power-on Mode

To initiate communication with the target using power-on mode, apply V_{DD} to the target, as shown in Figure 2-7. The target drives the SDATA line high. The programmer then waits and polls for a high-to-low transition on the SDATA line, which is the signal from the target that V_{DD} has stabilized. Note that until V_{DD} stabilizes, the SDATA signal is noisy and a false edge can be detected. As a result, the programmer must wait for the time specified by $T_{VDDWAIT}$ (Table 2-2) before beginning to wait and poll. The programmer must also not drive the SCLK signal until the $T_{VDDWAIT}$ time period has passed.

Figure 2-7. Acquire Sequence Timing using VDD Power-on Mode Acquire



Notes

- R0 = Resistive logic ‘0’ output
- HIZ = High-impedance with logic Input disabled
- S1 = Strong logic '1' output from the target
- The programmer must wait for the time specified by $T_{VDDWAIT}$ before beginning to wait and poll for a S1 to R0 transition

After the SDATA transition is detected, the programmer must transmit the ID-SETUP-1 vectors in T_{ACQ} seconds (see Table 2-2) and wait and poll for a high-to-low transition on SDATA.

After the ID-SETUP-1 mnemonic, send the ID-SETUP-2 mnemonic and wait and poll. Next, send the SYNC-ENABLE, READ-ID-WORD, and SYNC-DISABLE mnemonics. See Bit Streams for Mnemonics for mnemonic bit streams.

During the power cycle phase of the Initialize Target procedure, V_{DD} must be the only pin asserted. XRES must be low. The internal pull-down resistor of the CY8C20xx6A on XRES achieves this if the pin is left floating externally.

2.3.3 Power-on Mode with External Supply

Acquiring the device through the power-on mode is possible even if the device uses an external power supply. Be careful with the timing and voltage levels of the device and power supply. This is because the programmer does not have control over the power supply in this case.

The programmer must be able to detect the supply voltage to determine when the supply has reached the minimum programming level and nominal supply voltage (for example, 1.8 V, 3.3 V, or 5 V). As shown in Figure 2-7, the programming sequence timing is initiated when the supply voltage reaches 1.66 V (±3%). At this point, the programmer must wait for some time before starting the Wait-and-Poll procedure. As shown in Table 2-1, the high and low signal thresholds (VILP, VIHP) are dictated by the power supply level of the target. Note that it is possible to use 3.3 V signal levels even if the target is powered by a 5 V supply. However, it is not possible to use 3.3 V signal levels if the target is powered by a 1.8 V supply.

A particular area of concern is the power supply ramp rate of the target. For an extremely slow ramp rate (< 0.5 V/ms), it is possible that the supply voltage may not reach the nominal voltage before the programming mode acquisition window closes. In this case, the programmer should specify a minimum supply ramp rate or develop an intelligent way to track the supply voltage and dynamically change the signal levels to match the supply voltage.

2.3.4 Read-ID-Word

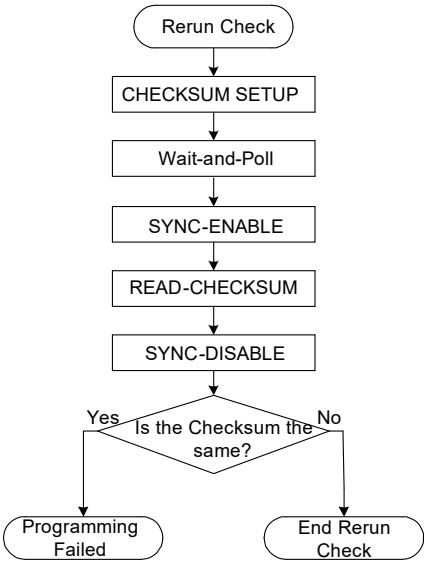
The silicon ID value is read back using the READ-ID-WORD vector-set. The first two bytes read back from the device for a READ-ID-WORD vector contain the silicon ID. The vectors in Bit Streams for Mnemonics under READ-ID-WORD show the device-specific values read from the target. For example, a LLLLLLLL, HLLHHLHL denotes a 0x009A hex read back from a CY8C20066. The programmer must compare the value in the READ-ID-VECTOR and the value returned by the target. If these values do not match, the programmer must terminate the programming flow.

2.4 Rerun Check (Optional)

The Rerun Check procedure compares the device flash checksum with the hex file to see if the chip is previously programmed. When this feature is enabled, the chip inside a socket fails when it is programmed a second time. The rerun check is an optional feature for production programmers connected to a handler.

The rerun check is accomplished by the sequence shown in Figure 2-8. The CHECKSUM-SETUP, SYNC-ENABLE, READ-CHECKSUM, and SYNC-DISABLE bit streams are shown in Bit Streams for Mnemonics. See Wait-and-Poll for detailed timing information.

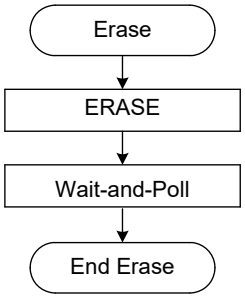
Figure 2-8. Rerun Check Procedure



2.5 Erase

The Erase procedure erases the entire flash memory and security data (all set to zero). Erase is accomplished by sending the ERASE vector followed by the Wait-and-Poll procedure (see Figure 2-9).

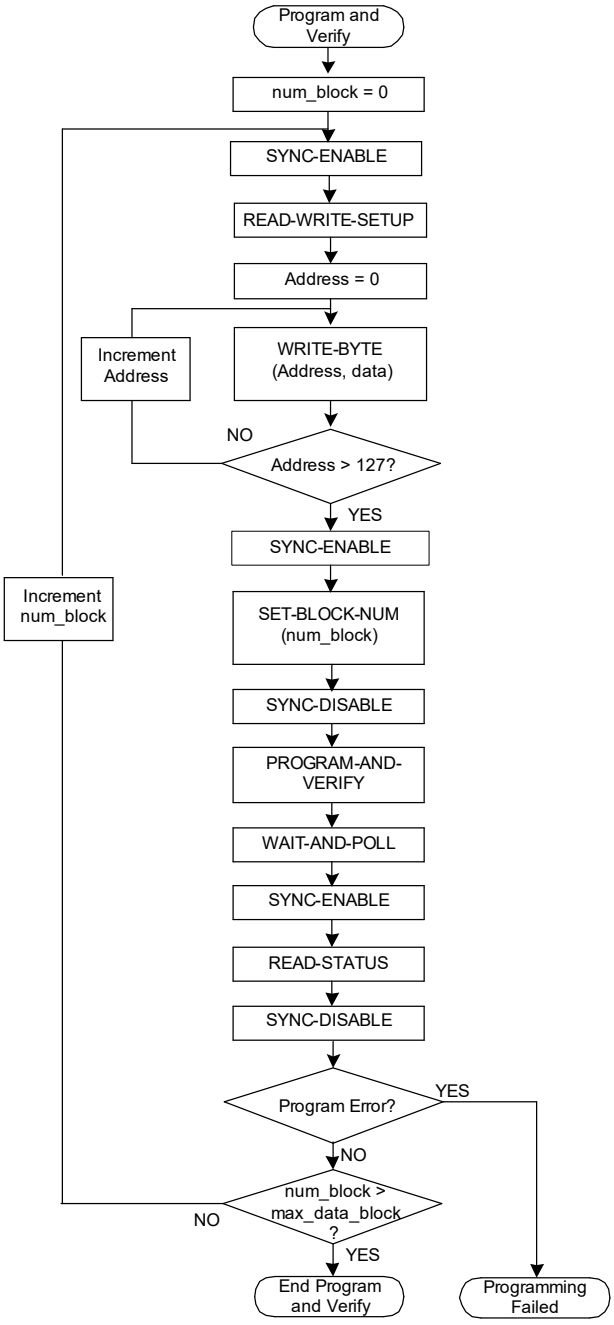
Figure 2-9. Erase Procedure



2.6 Program and Verify

The Program and Verify procedure programs the flash with the contents of the user's programming file. The READ-STATUS vector is used to determine the pass/fail success of the PROGRAM-AND-VERIFY vector. A programming failure results in a 0x04 value and a pass results in a 0x00 value. Figure 2-10 shows the sequence for the Program and Verify procedure.

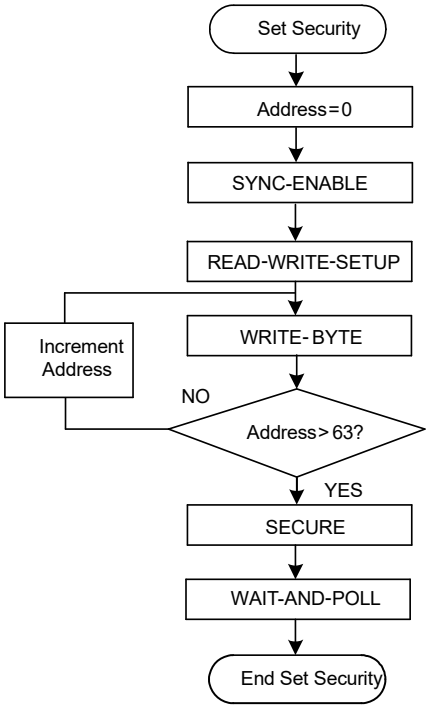
Figure 2-10. Program and Verify Procedure



2.7 Set Security

The Set Security procedure is performed after the device is successfully programmed. The SECURE mnemonic protects certain flash blocks from being read or changed. The security data for each block is located at the end of the hex file; see Bit Streams for Mnemonics for format details. The Set Security procedure is shown in Figure 2-11.

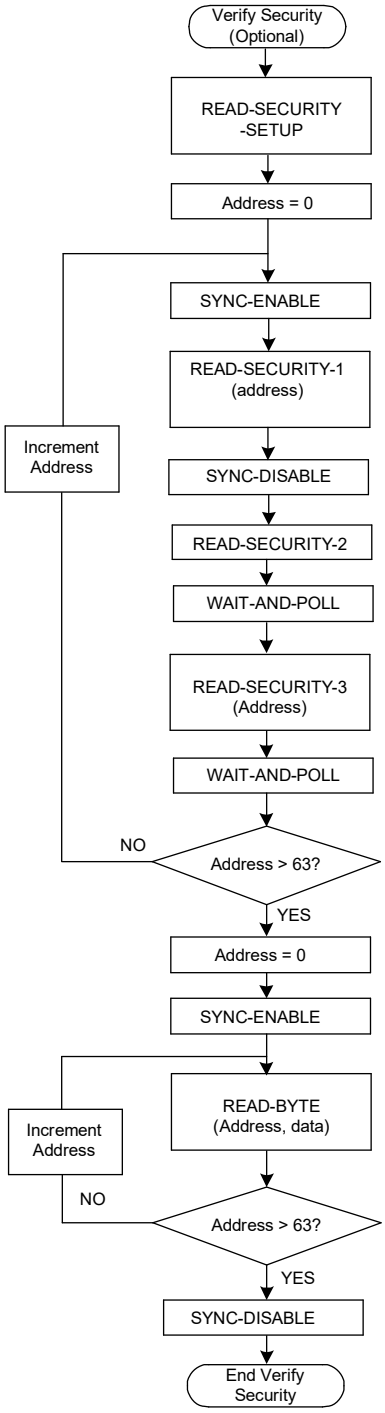
Figure 2-11. Set Security Procedure



2.8 Verify Security (Optional)

In the optional Verify Security procedure, the programmer reads the security data from the chip and stores it in the memory. This data is compared with the programming file or the security data used in the Set Security step. The Verify Security procedure is shown in Figure 2-12.

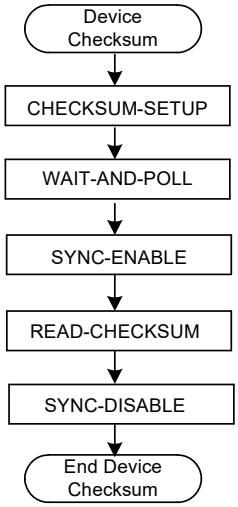
Figure 2-12. Verify Security Procedure



2.9 Device Checksum

In the Device Checksum procedure, the checksum is retrieved from the device and compared to the device checksum set in the target file. Note that the device checksum is not the same as the 'record checksum'. The Device Checksum procedure is shown in Figure 2-13.

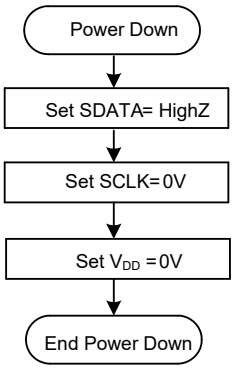
Figure 2-13. Device Checksum Procedure



2.10 Power Down

The last step in the programming data flow is to power down the device. Power down is accomplished by the sequence shown in Figure 2-14.

Figure 2-14. Power Down Procedure

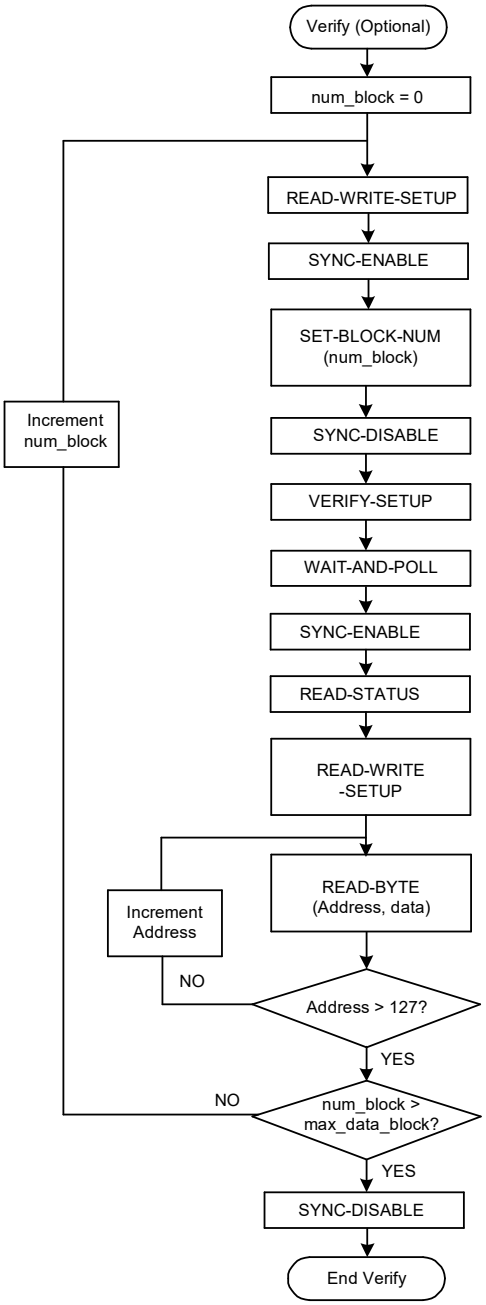


2.11 Verify (Optional)

The Verify procedure reads data from the device's flash so that it can be compared to the programming file. The READ-STATUS vector is used to determine whether the block is protected. The READ-STATUS vector returns a 0x01 for a secured block and the actual data read from that block is invalid. In the case of a secured block, the programmer should display 'XX' for the data in the block.

Note that the Program and Verify procedure already performs a verify check between the flash and programming file. Therefore, it is not necessary to perform the optional Verify procedure. The max_data_block value for each device is listed in Table 2-3. The Verify procedure is shown in Figure 2-15.

Figure 2-15. Verify Procedure



2.12 Specifications and Definitions

2.12.1 DC Programming Specifications

Note These specifications are from the CY8C20xx6A datasheet. To view the complete data sheet visit the Cypress web site.

Table 2-1. DC Programming Specifications

DC Programming Specifications	Minimum	Typical	Maximum[a]	Maximum[b]
V_{DDP} (V_{DD} for programming and erase)	1.71 V	3.3 V	5.5 V	5.25 V
I_{DDP} (Supply current during programming or verify)	–	5.0 mA	25 mA	25 mA
V_{ILP} (Input low voltage during programming or verify) $V_{DDP} \geq 2.6$ V	–0.3 V	0 V	0.8 V	0.8 V
V_{IHP} (Input high voltage during programming or verify)	1.71 V	V_{DDP}	V_{DDP} + 0.3 V	V_{DDP} + 0.3 V
I_{ILP} (Input current when applying V_{ILP} to P1[0] or P1[1] during programming or verify)	–	–	0.2 mA	0.2 mA
I_{IHP} (Input current when applying V_{IHP} to Port Pins during programming or verify)	–	–	1.5 mA	1.5 mA
V_{OLP} (Output low voltage during programming or verify, I_{OL} = 0.1 mA)	–	–	V_{SS} + 0.75 V	V_{SS} + 0.75 V
V_{OHP} (Output high voltage during programming or verify I_{OH}=5 mA)	V_{DDP} – 0.9 V	–	V_{DDP}	V_{DDP}

a. Maximum for Silicon Rev B3 Devices
b. Maximum for Silicon Rev E1 Devices

2.12.2 AC Programming Specifications

Table 2-2. AC Programming Specifications

AC Programming Specifications	Minimum	Maximum
T_{XRES} (XRES pulse length[a])	300 µs	_
$T_{VDDWAIT}$ (V_{DD} stable to Wait-and-Poll hold off [b])	0.1 ms	1 ms
$T_{VDDXRES}$ (V_{DD} stable to XRES assertion delay)	14.27 ms	_
T_{POLL} (SDATA high pulse time[c])	10 µs	200 ms
T_{ACQ} ("Key window" time after a V_{DD} ramp acquire event, based on 256 ILO clocks.[d],[e])	3.2 ms	19.6 ms
$T_{XRESACQ}$ ("Key window" time after an XRES event, based on 8 ILO clocks.[e])	98 µs	615 µs
T_{RSCLK} (Rise time of SCLK)	1 ns	20 ns
T_{FSCLK} (Fall time of SCLK)	1 ns	20 ns
T_{SSCLK} (Data setup time to falling edge of SCLK)	40 ns	_
T_{HSCLK} (Data hold time from falling edge of SCLK)	40 ns	_
F_{SCLK} (Frequency of SCLK)	_	8 MHz[f]
T_{DSCLK} (Data-out delay from falling edge of SCLK (3.6 V < V_{DD})	_	60 ns[f]
T_{DSCLK3} (Data-out delay from falling edge of SCLK (3.0 V ≤ V_{DD} ≤ 3.6 V)	_	85 ns[f]
T_{DSCLK2} (Data-out delay from falling edge of SCLK (1.71 V ≤ V_{DD} ≤ 3.0 V)	_	130 ns[f]

a. See Figure 2-6. Times longer than T_{XRES} may be used without consequence.
b. Until V_{DD} stabilizes, SDATA is noisy and the falling edge should not be searched for. For this reason, a delay of $T_{VDDWAIT}$ is necessary after V_{DD} is applied and before Wait-and-Poll.
c. T_{POLL} applies to the Wait-and-Poll procedure. SDATA remains high for the T_{POLL} time.
d. The ID-SETUP-1 bit stream data must not be delayed more than T_{ACQ} from the end of the Wait-and-Poll (measured from the falling edge of SDATA).
e. The "key window" is the length of time during which a specific bit sequence or "key" must be sent to the target device, on SDATA.
f. With capacitance ≤ 30 Pico Farad.

2.12.3 Device Address and Block Definitions

Table 2-3. Device Address and Block Definitions

Device	Address numbers for bytes within a block	Block numbers for program data	max_data_block
CY7C60400	0-127	0-255	255
CY7C60413	0-127	0-63	63
CY7C60445	0-127	0-127	127
CY7C60455	0-127	0-127	127
CY7C60456	0-127	0-255	255
CY7C64300	0-127	0-255	255
CY7C64313	0-127	0-63	63
CY7C64315	0-127	0-127	127
CY7C64316	0-127	0-255	255
CY7C64343	0-127	0-63	63
CY7C64345	0-127	0-127	127
CY7C64355	0-127	0-127	127
CY7C64356	0-127	0-255	255
CY8C20045-24LKXI	0-127	0-63	63
CY8C20055-24LKXI	0-127	0-63	63
CY8C20055-24SXI	0-127	0-63	63
CY8C20066	0-127	0-255	255
*** CY8C20236-24LKXI	0-127	0-63	63
*** CY8C20246-24LKXI	0-127	0-127	127
*** CY8C20266-24LKXI	0-127	0-255	255
*** CY8C20336-24LQXI	0-127	0-63	63
CY8C20336H-24LQXI	0-127	0-63	63
*** CY8C20346-24LQXI	0-127	0-127	127
CY8C20346H-24LQXI	0-127	0-127	127
*** CY8C20366-24LQXI	0-127	0-255	255
*** CY8C20396-24LQXI	0-127	0-255	255
*** CY8C20436-24LQXI	0-127	0-63	63
*** CY8C20446-24LQXI	0-127	0-127	127
CY8C20446H-24LQXI	0-127	0-127	127
*** CY8C20466-24LQXI	0-127	0-255	255
*** CY8C20496-24LQXI	0-127	0-127	127
*** CY8C20536 -24PVXI	0-127	0-63	63
*** CY8C20546 -24PVXI	0-127	0-127	127
*** CY8C20566 -24PVXI	0-127	0-255	255
*** CY8C20636-24LQXI	0-127	0-63	63
*** CY8C20646-24LQXI	0-127	0-127	127
*** CY8C20666-24LQXI	0-127	0-255	255
CY8CTST200-16LGXI	0-127	0-255	255
CY8CTST200-24LQXI	0-127	0-255	255
CY8CTST200-32LQXI	0-127	0-255	255

Table 2-3. Device Address and Block Definitions

CY8CTST200-48LTXI	0-127	0-255	255
CY8CTST200-48PVXI	0-127	0-255	255
CY8CTMG200-00LGXI	0-127	0-255	255
CY8CTMG200-16LGXI	0-127	0-255	255
CY8CTMG200-24LQXI	0-127	0-255	255
CY8CTMG200-32LQXI	0-127	0-255	255
CY8CTMG200-48LTXI	0-127	0-255	255
CY8CTMG200-48PVXI	0-127	0-255	255
CY8CTMG201-32LQXI	0-127	0-127	127
CY8CTMG201-48LTXI	0-127	0-127	127
CY8CTMG201-48PVXI	0-127	0-127	127
CY8C20066A	0-127	0-255	255
*** CY8C20236A-24LKXI	0-127	0-63	63
*** CY8C20246A-24LKXI	0-127	0-127	127
*** CY8C20246AS-24LKXI	0-127	0-127	127
*** CY8C20266A-24LKXI	0-127	0-255	255
*** CY8C20336A-24LQXI	0-127	0-63	63
*** CY8C20336AN-24LQXI	0-127	0-63	63
*** CY8C20346A-24LQXI	0-127	0-127	127
*** CY8C20346AS-24LQXI	0-127	0-127	127
*** CY8C20366A-24LQXI	0-127	0-255	255
*** CY8C20396A-24LQXI	0-127	0-255	255
*** CY8C20436A-24LQXI	0-127	0-63	63
*** CY8C20436AN-24LQXI	0-127	0-63	63
*** CY8C20446A-24LQXI	0-127	0-127	127
*** CY8C20446AS-24LQXI	0-127	0-127	127
*** CY8C20446L-24LQXI	0-127	0-127	127
*** CY8C20466A-24LQXI	0-127	0-255	255
*** CY8C20466AS-24LQXI	0-127	0-255	255
*** CY8C20466L-24LQXI	0-127	0-255	255
*** CY8C20496A-24LQXI	0-127	0-127	127
*** CY8C20496L-24LQXI	0-127	0-127	127
*** CY8C20536A -24PVXI	0-127	0-63	63
*** CY8C20546A -24PVXI	0-127	0-127	127
*** CY8C20546L -24PVXI	0-127	0-127	127
*** CY8C20566A -24PVXI	0-127	0-255	255
*** CY8C20566L -24PVXI	0-127	0-255	255
*** CY8C20636A-24LTXI CY8C20636A-24LQXI	0-127	0-63	63
*** CY8C20636AN-24LTXI	0-127	0-63	63
*** CY8C20646A-24LTXI CY8C20646A-24LQXI	0-127	0-127	127
*** CY8C20646AS-24LTXI	0-127	0-127	127
*** CY8C20646L-24LTXI CY8C20646L-24LQXI	0-127	0-127	127

Table 2-3. Device Address and Block Definitions

*** CY8C20666A-24LTXI CY8C20666A-24LQXI	0-127	0-255	255
*** CY8C20666AS-24LTXI	0-127	0-255	255
*** CY8C20666L-24LTXI CY8C20666L-24LQXI	0-127	0-255	255
*** CY8C20746A 24FDXC	0-127	0-127	127
*** CY8C20766A 24FDXC	0-127	0-255	255
CY8CTST200A-16LGXI	0-127	0-255	255
CY8CTST200A-24LQXI	0-127	0-255	255
CY8CTST200A-32LQXI	0-127	0-255	255
CY8CTST200A-48LTXI	0-127	0-255	255
CY8CTST200A-48PVXI	0-127	0-255	255
CY8CTST241-LQI-01	0-127	0-255	255
CY8CTST241-LTI-01	0-127	0-255	255
CP8CTST241-FNC-01	0-127	0-255	255
CY8CTMG200A-00LGXI	0-127	0-255	255
CY8CTMG200A-16LGXI	0-127	0-255	255
CY8CTMG200A-24LQXI	0-127	0-255	255
CY8CTMG200A-32LQXI	0-127	0-255	255
CY8CTMG200A-48LTXI	0-127	0-255	255
CY8CTMG200AH-48LTXI	0-127	0-255	255
CY8CTMG200A-48PVXI	0-127	0-255	255
CY8CTMG201A-32LQXI	0-127	0-127	127
CY8CTMG201A-48LTXI	0-127	0-127	127
CY8CTMG201A-48PVXI	0-127	0-127	127
CY8CTMG240-LQI-01	0-127	0-255	255
CY8CTMG240-LTI-01	0-127	0-255	255
CP8CTMG240-FNC-01	0-127	0-255	255
CY8C20237-24SXI	0-127	0-63	63
CY8C20247-24SXI	0-127	0-127	127
CY8C20237-24LKXI	0-127	0-63	63
CY8C20247-24LKXI	0-127	0-127	127
CY8C20247S-24LKXI	0-127	0-127	127
CY8C20337-24LQXI	0-127	0-63	63
CY8C20337AN-24LQXI	0-127	0-63	63
CY8C20347-24LQXI	0-127	0-127	127
CY8C20347S-24LQXI	0-127	0-127	127
CY8C20437-24LQXI	0-127	0-63	63
CY8C20437AN-24LQXI	0-127	0-63	63
CY8C20447-24LQXI	0-127	0-127	127
CY8C20447S-24LQXI	0-127	0-127	127
CY8C20467-24LQXI	0-127	0-255	255
CY8C20467S-24LQXI	0-127	0-255	255
CY8C20637-24LQXI	0-127	0-63	63
CY8C20637AN-24LQXI	0-127	0-63	63

Table 2-3. Device Address and Block Definitions

CY8C20647-24LQXI	0-127	0-127	127
CY8C20647S-24LQXI	0-127	0-127	127
CY8C20667-24LQXI	0-127	0-255	255
CY8C20667S-24LQXI	0-127	0-255	255
CY8C20747-24FDXC	0-127	0-127	127
CY8C20767-24FDXC	0-127	0-255	255

A. Programming Mnemonics for CY8C20045, CY8C20055, CY8C20xx6, CY8C20xx7, CY7C643xx, and CY7C604xx

A.1 Bit Streams for Mnemonics

(**Note** Use caution when copying these bit streams from Adobe Reader. Issues have been seen where bits are dropped during the Copy and Paste process. The bit streams below have been organized into 66-bit chunks to help prevent this problem.)

```
ID-SETUP-1
11001010000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000
11011110111000100001111101111111000000100111110111000100000010011 1
11011100000000000011111110111101110000000011110011111000001110101 11
10011111001000000111111001111101100010000111110111101110001000011 1
11011111110000000001111101110001000000000111110111000000000000111 11
11011110111000000001111101111010000000011111110111101010000000011 1
11011110110000000001111101111000000000100111111011111000000000000 111
11011111001010000001111101111101000110000111110111111110001001011 0

ID-SETUP-2
11011110111000100001111101111111000000100111110111000100000010011 1
11011100000000000011111100111111000001110101111001111001000000111 11
10011111010000000001111101111111000000000111110111000100000000011 1
11011100000000000011111110111101110000000011111011111010000000011 111
11011110101000000001111101111011000000000111110111110000000000110 111
11011111000000000001111101111100101000000111110111111010001100001 11
11011111111000100101 10

SET-BLOCK-NUM
10011111010ddddddd111
      dddddddd is block number.

CHECKSUM-SETUP
11011110111000100001111101111111000000100111110111000100000010011 1
11011100000000000011111100111111000001110101111001111001000000111 11
10011111010000000001111101111111000000000111110111000100000000011 1
11011100000000000011111110111101110000000011111011111010000000011 111
11011110101000000001111101111011000000000111110111110000000000111 111
11011111000000000001111101111100101000000111110111111010001100001 11
11011111111000100101 10
```

```
READ-CHECKSUM
10111111001ZZDDDDDDDD1
10111111000ZZddddddddd1
      DDDDDDDD is Device Checksum upper byte (MSB)
      dddddddd is Device Checksum lower byte (LSB)

PROGRAM-AND-VERIFY
11011110111000100001111101111111000000100111110111000100000010011
11011100000000000111111001111110000011101011110011111001000000111
10011111011100000001111101111111000000000111110111000100000000111
11011100000000000111111011110111000000001111101101010000000001111
11011110100000000111111011110101000000001111101111011000000000111
11011110000000010101111101111100000000000111110111110010100000111
11011111010001100001111101111111100010010110

ERASE
11011110111000100001111101111111000000100111110111000100000010011
11011100000000000111111001111110000011101011110011111001000010111
11011111110000000001111101110001000000000111110111000000000000111
11011110111000000001111101111000000000010111111011110100000000111
11011110101000000001111101111011000001000111110111111000000000111
11011111001001100001111101111101001000000111110111111110001001011
0

SECURE
11011110111000100001111101111111000000100111110111000100000010011
11011100000000000111111001111110000011101011110011111001000000111
10011111011100000001111101111111000000000111110111000100000000111
11011100000000000111111011110111000000001111101101010000000001111
11011110100000000111111011110101000000001111101111011000000000111
11011110000000010011111101111100000000000111110111110010100000111
11011111010001100001111101111111100010010110

READ-SECURITY-SETUP
11011110111000100001111101100000100010000111110110000100001000011
11011110111000000000111

READ-SECURITY-1
11011110111000100001111101110010100001110111110111001010000000111
110111001010aaaaaaa1111101110010100000000111
      aaaaaaa = address (7 bits)

READ-SECURITY-2
11011110111000000001111101111010000000011111110111101010000000111
11011110110000000001111101111100001011101111110111110011110010111
11011111010001100001111101111011000100001111101111111100010010110
```

```
READ-SECURITY-3
1101111011100000000111110111101000000001111111011110101000000000111
1101111011000000000111110111110000101001111111011111001laaaaaaa111
110111111010001100001111101111111100010010110
      aaaaaaa = address (7 bits)

READ-WRITE-SETUP
1101111011110000000111110111100000000000011111011010000000000001111

WRITE-BYTE
1001aaaaaaaddddddddd111
      aaaaaaa = address (7 bits), dddddddd = data in

READ-ID-WORD (CY8C20066 & CY8C20066A)
101111111000ZZLLLLLLLL110111111001ZZHLLHHLHL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20236 & CY8C20236A)
101111111000ZZLLLLLLLL110111111001ZZHLHHLLHH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20246 & CY8C20246A)
101111111000ZZLLLLLLLL110111111001ZZHLHLHLHL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20266 & CY8C20266A)
101111111000ZZLLLLLLLL110111111001ZZHLLHLHHL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20336 & CY8C20336A)
101111111000ZZLLLLLLLL110111111001ZZHLHHLHLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20336H-24LQXI)
101111111000ZZLLLLHHLL110111111001ZZHLHHLHLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20346 & CY8C20346A)
101111111000ZZLLLLLLLL110111111001ZZHLHLHHLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20346H-24LQXI)
101111111000ZZLLLLHHLL110111111001ZZHLHLHHLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20366 & CY8C20366A)
101111111000ZZLLLLLLLL110111111001ZZHLLHLHHH111111110011ZZLHLHLLHL
```

```
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20396 & CY8C20396A)
10111111000ZZLLLLLLLL110111111001ZZHLHLHHHH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20436 & CY8C20436A)
10111111000ZZLLLLLLLL110111111001ZZHLHHLHLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20446 CY8C20446A)
10111111000ZZLLLLLLLL110111111001ZZHLHLHHLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20446H-24LQXI)
10111111000ZZLLLLHHLL110111111001ZZHLHLHHLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20466 & CY8C20466A)
10111111000ZZLLLLLLLL110111111001ZZHLLHHLLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20496 & CY8C20496A)
10111111000ZZLLLLLLLL110111111001ZZHLHHHHLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20536 & CY8C20536A)
10111111000ZZLLLLLLLL110111111001ZZHLHHHLLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20546 & CY8C20546A)
10111111000ZZLLLLLLLL110111111001ZZHLHLHHHL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20566 & CY8C20566A)
10111111000ZZLLLLLLLL110111111001ZZHLLHHLLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20636 & CY8C20636A)
10111111000ZZLLLLLLLL110111111001ZZHLHHHLHL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20646 & CY8C20646A)
10111111000ZZLLLLLLLL110111111001ZZHLHHHLLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1
```

READ-ID-WORD (CY8C20666 & CY8C20666A)
10111111000ZZLLLLLLLL110111111001ZZHLLHHHLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20746A)
10111111000ZZLLLLLLLL110111111001ZZHLHHHHHL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20766A)
10111111000ZZLLLLLLLL110111111001ZZHLHHHHHH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY7C60400)
10111111000ZZLLLLLLLL110111111001ZZHLHLHLHH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY7C60413)
10111111000ZZLLLLLLLL110111111001ZZHLHHLHHL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY7C60445)
10111111000ZZLLLLLLLL110111111001ZZHLHLLHHH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY7C60455)
10111111000ZZLLLLLLLL110111111001ZZHLHLHLLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY7C60456)
10111111000ZZLLLLLLLL110111111001ZZHLHLHLLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY7C64300)
10111111000ZZLLLLLLLL110111111001ZZHLHLLHHL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (cy7c64315)
10111111000ZZLLLLLLLL110111111001ZZHLHLLLLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (cy7c64316)
10111111000ZZLLLLLLLL110111111001ZZHLHLLLLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY7C64343)
10111111000ZZLLLLLLLL110111111001ZZHLHHLHHH111111110011ZZLHLHLLHL

```
111111110000ZZLLHLLLLH1

READ-ID-WORD (cy7c64345)
10111111000ZZLLLLLLLL110111111001ZZHLHLLLHL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (cy7c64355)
10111111000ZZLLLLLLLL110111111001ZZHLHLLLHH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (cy7c64356)
10111111000ZZLLLLLLLL110111111001ZZHLHLLHLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTMG200-00LTXI & CY8CTMG200A-00LTXI)
10111111000ZZLLLLLHHH110111111001ZZHLLHHLHL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTMG200-16LGXI & CY8CTMG200A-16LGXI)
10111111000ZZLLLLLHHH110111111001ZZLHHLHHLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTMG200-24LQXI & CY8CTMG200A-24LQXI)
10111111000ZZLLLLLHHH110111111001ZZLHHLHHLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTMG200-32LQXI & CY8CTMG200A-32LQXI)
10111111000ZZLLLLLHHH110111111001ZZLHHLHHHL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTMG200-48LTXI & CY8CTMG200A-48LTXI)
10111111000ZZLLLLLHHH110111111001ZZLHHLHHHH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTMG200AH-48LTXI)
10111111000ZZLLLLHHHH110111111001ZZLHHLHHHH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTMG240-LQI-01)
10111111000ZZLLLLLHHH110111111001ZZHHLHLHLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTMG240-LTI-01)
10111111000ZZLLLLLHHH110111111001ZZHHLHLHLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CP8CTMG240-FNC-01)
```

```
10111111000ZZLLLLLHHH110111111001ZZHLHHHHHH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTMG200-48PVXI & CY8CTMG200A-48PVXI)
10111111000ZZLLLLLHHH110111111001ZZLLLHHHLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTST200-16LGXI & CY8CTST200A-16LGXI)
10111111000ZZLLLLLHHL110111111001ZZLHHLHHLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTST200-24LQXI & CY8CTST200A-24LQXI)
10111111000ZZLLLLLHHL110111111001ZZLHHLHHLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTST200-32LQXI & CY8CTST200A-32LQXI)
10111111000ZZLLLLLHHL110111111001ZZLHHLHHHL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTST200-48LTXI & CY8CTST200A-48LTXI)
10111111000ZZLLLLLHHL110111111001ZZLHHLHHHH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTST200-48PVXI & CY8CTST200A-48PVXI)
10111111000ZZLLLLLHHL110111111001ZZLLLHHHLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTST241-LQI-01)
10111111000ZZLLLLLHHL110111111001ZZHHLHLHHL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTST241-LTI-01)
10111111000ZZLLLLLHHL110111111001ZZHHLHLHHH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CP8CTST241-FNC-01)
10111111000ZZLLLLLHHL110111111001ZZHLHHHHHH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTMG201-16LQXI)
10111111000ZZLLLLLHHH110111111001ZZLHHLLLLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTMG201-24LQXI)
10111111000ZZLLLLLHHH110111111001ZZLHHLLLHL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1
```

READ-ID-WORD (CY8CTMG201-32LQXI)
10111111000ZZLLLLLHHH110111111001ZZLHHLLLHH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTMG201-48LTXI & CY8CTMG201A-48LTXI)
10111111000ZZLLLLLHHH110111111001ZZLHHLLHLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTMG201-48PVXI & CY8CTMG201A-48PVXI)
10111111000ZZLLLLLHHH110111111001ZZLHHLLHLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CP8CTST242-FNC-01)
10111111000ZZLLLLLHHL110111111001ZZHLHHHHHL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20246AS-24LKXI)
10111111000ZZLLLLHLHH110111111001ZZHLHLHLHL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20336AN-24LQXI)
10111111000ZZLLLLLLLL110111111001ZZHHLHHLHH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20346AS-24LQXI)
10111111000ZZLLLLHLHH110111111001ZZHLHLHHLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20436AN-24LQXI)
10111111000ZZLLLLLLLL110111111001ZZHHLHHHLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20446AS-24LQXI)
10111111000ZZLLLLHLHH110111111001ZZHLHLHHLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20446L-24LQXI)
10111111000ZZLLLLLLHH110111111001ZZHLHLHHLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20466AS-24LQXI)
10111111000ZZLLLLHLHH110111111001ZZHLLHHLLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20466L-24LQXI)
10111111000ZZLLLLLLHH110111111001ZZHLLHHLLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20496L-24LQXI)
10111111000ZZLLLLLLHH110111111001ZZHLHHHHLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20546L-24PVXI)
10111111000ZZLLLLLLHH110111111001ZZHLHLHHHL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20566L-24PVXI)
10111111000ZZLLLLLLHH110111111001ZZHLLHHLLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20636AN-24LTXI)
10111111000ZZLLLLLLLL110111111001ZZHHLHHLLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20646AS-24LQXI)
10111111000ZZLLLLHLHH110111111001ZZHLHHHLLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20646AS-24LTXI)
10111111000ZZLLLLHLHH110111111001ZZHLHHHLLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

*** READ-ID-WORD (CY8C20646L-24LTXI & CY8C20646L-24LQXI)
10111111000ZZLLLLLLHH110111111001ZZHLHHHLLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20666AS-24LQXI)
10111111000ZZLLLLHLHH110111111001ZZHLLHHHLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20666AS-24LTXI)
10111111000ZZLLLLHLHH110111111001ZZHLLHHHLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

*** READ-ID-WORD (CY8C20666L-24LTXI CY8C20666L-24LQXI)
10111111000ZZLLLLLLHH110111111001ZZHLLHHHLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTMG201-16LGXI)
10111111000ZZLLLLLHHH110111111001ZZLHHLLLLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTMG201-24LQXI)
10111111000ZZLLLLLHHH110111111001ZZLHHLLLHL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

```
READ-ID-WORD (CY8CTST201-16LGXI)
10111111000ZZLLLLLHHL110111111001ZZLHHLLLLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTST201-24LQXI)
10111111000ZZLLLLLHHL110111111001ZZLHHLLLHL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTST201-32LQXI)
10111111000ZZLLLLLHHL110111111001ZZLHHLLLHH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTST201-48LTXI)
10111111000ZZLLLLLHHL110111111001ZZLHHLLHLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTST201-48PVXI)
10111111000ZZLLLLLHHL110111111001ZZLHHLLHLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTST242-LQI-01)
10111111000ZZLLLLLHHL110111111001ZZHLLHHLLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8CTST242-LTI-01)
10111111000ZZLLLLLHHL110111111001ZZHLLHHHLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20237-24LKXI)
10111111000ZZLLLLLLLH110111111001ZZLHLLLLHL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20237-24SXI)
10111111000ZZLLLLLLLH110111111001ZZLHLLLLLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20247-24LKXI)
10111111000ZZLLLLLLLH110111111001ZZLHLLLLHH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20247S-24LKXI)
10111111000ZZLLLLHLHH110111111001ZZLHLLLLHH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20247-24SXI)
10111111000ZZLLLLLLLH110111111001ZZLHLLLLLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1
```

READ-ID-WORD (CY8C20337-24LQXI)
10111111000ZZLLLLLLLH110111111001ZZLHLLLHLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20337AN-24LQXI)
10111111000ZZLLLLLLLH110111111001ZZLHLHLLLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20347-24LQXI)
10111111000ZZLLLLLLLH110111111001ZZLHLLLHLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20347S-24LQXI)
10111111000ZZLLLLHLHH110111111001ZZLHLLLHLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20437-24LQXI)
10111111000ZZLLLLLLLH110111111001ZZLHLLLHHL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20437AN-24LQXI)
10111111000ZZLLLLLLLH110111111001ZZLHLHLLLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20447-24LQXI)
10111111000ZZLLLLLLLH110111111001ZZLHLLLHHH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20447S-24LQXI)
10111111000ZZLLLLHLHH110111111001ZZLHLLLHHH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20467-24LQXI)
10111111000ZZLLLLLLLH110111111001ZZLHLLHLLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20467S-24LQXI)
10111111000ZZLLLLHLHH110111111001ZZLHLLHLLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20637-24LQXI)
10111111000ZZLLLLLLLH110111111001ZZLHLLHLLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20637AN-24LQXI)
10111111000ZZLLLLLLLH110111111001ZZLHLLHHHH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

```
READ-ID-WORD (CY8C20647-24LQXI)
10111111000ZZLLLLLLLH110111111001ZZLHLLHLHL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20647S-24LQXI)
10111111000ZZLLLLHLHH110111111001ZZLHLLHLHL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20667-24LQXI)
10111111000ZZLLLLLLLH110111111001ZZLHLLHLHH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20667S-24LQXI)
10111111000ZZLLLLHLHH110111111001ZZLHLLHLHH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20747-24FDXC)
10111111000ZZLLLLLLLH110111111001ZZLHLLHHLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20767-24FDXC)
10111111000ZZLLLLLLLH110111111001ZZLHLLHHLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20045-24LKXI)
10111111000ZZLLLLLLLH110111111001ZZHLHLLHLH111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20055-24LKXI)
10111111000ZZLLLLLLLH110111111001ZZLHHHLLLL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

READ-ID-WORD (CY8C20055-24SXI)
10111111000ZZLLLLLLLH110111111001ZZHHLLHHHL111111110011ZZLHLHLLHL
111111110000ZZLLHLLLLH1

VERIFY-SETUP
1101111011100010000111110111111100000010011111011100010000000100111
1101110000000000001111110011111000001110101111001111100100000011111
1001111101110000000111110111111100000000011111011100010000000000111
1101110000000000001111111011110111000000001111101101010100000001111
1101111010000000011111110111101010000000011111011111011000000000111
1101111000000000001111110111110000000000011111011111001010000000111
1101111101000110000111110111111100010010110

READ-STATUS
10111111000ZZDDDDDDDD1
      DDDDDDDD = data out
```

```
READ-BYTE
1011aaaaaaaZZDDDDDDDD1
      aaaaaaa = address (7 bits), DDDDDDDD = data out

SYNC-ENABLE
1101111011100010000111110111111100000010011111011100010000000100111
1101111000000000001111111011110111000000000111

SYNC-DISABLE
1101111011100010000111110111000100000000011111011111110000000000111
1101111000000000001111111011110111000000000111
```

Notes: 1=logic high=Vihp
0=logic low=Vilp
Z=High Z (floating)
D=data read from device (Most Significant Bit of the binary data comes out first)
d=data applied to the device (MSB of the binary data goes in first)
a=address applied to the device (MSB of the binary data goes in first)
H=High data read from the device (Vout=Vohv)
L=Low data read from the device (Vout=Volv)
If the programmer has delays between executing the different Mnemonics, SDATA should be HighZ (floating).

Other Mnemonics

WAIT-AND-POLL:

After the mnemonic bit stream is sent, the SDATA pin typically takes 1 uS to drift low. (The SDATA pin drifts low to Vilp by the device's internal pull down resistor.) A minimum delay of 416 ns (or ten 12-MHz CPU clock cycles) is needed before the SDATA pin is pulled high. Clocking is needed in order for SDATA to transition from low to high. At least one SCLK should be sent, but more may be sent before SDATA to transition from low to high. This is accomplished through IOWC at the end of the mnemonics. The target device pulls SDATA high when the mnemonic begins executing. No SCLKs are required while SDATA is high.

The device outputs a logic high on the SDATA pin while the mnemonic is executing and then switches to a logic low when the mnemonic finishes. The programmer should Wait & Poll the SDATA pin for the high-to-low transition. The maximum SDATA high time is 200 ms.

Once the transition to low is observed, the programmer should apply a bit stream of 0000000000 0000000000 0000000000 (30 zero bits) to the SDATA pin of the device and then continue to the next mnemonic.