zilog
Embedded in Life
An IXYS Company

Z8051 Series 8-Bit Microcontrollers

Z51F3220 MCU

Programming Specification

PRS001501-1013

PRELIMINARY

www.zilog.com

Warning: DO NOT USE THIS PRODUCT IN LIFE SUPPORT SYSTEMS.

LIFE SUPPORT POLICY

ZILOG'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS PRIOR WRITTEN APPROVAL OF THE PRESIDENT AND GENERAL COUNSEL OF ZILOG CORPORATION.

AS USED HEREIN

Life support devices or systems are devices which (a) are intended for surgical implant into the body, or (b) support or sustain life and whose failure to perform when properly used in accordance with instructions for use provided in the labeling can be reasonably expected to result in a significant injury to the user. A critical component is any component in a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system or to affect its safety or effectiveness.

DOCUMENT DISCLAIMER

 Information in this publication concerning the devices, applications, or technology described is intended to suggest possible uses and may be superseded. ZILOG, INC. DOES NOT ASSUME LIABILITY FOR OR PROVIDE A REPRESENTATION OF ACCURACY OF THE INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED IN THIS DOCUMENT. ZILOG ALSO DOES NOT ASSUME LIABILITY FOR INTELLECTUAL PROPERTY INFRINGEMENT RELATED IN ANY MANNER TO USE OF INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED HEREIN OR OTHERWISE. The information contained within this document has been verified according to the general principles of electrical and mechanical engineering.

ZMOTION, Z8 Encore! XP and eZ8 are trademarks or registered trademarks of Zilog, Inc. (An IXYS Company). All other product or service names are the property of their respective owners.

Revision History

Each instance in the following revision history table reflects a change to this document from its previous version. For more details, refer to the corresponding pages or appropriate links provided in the table.

Date	Revision Level	Description	Page Number
Oct 2013	01	Original issue.	All

Table of Contents

List of Figures

List of Tables

Overview

This document provides specifications for programming the Z51F3220 device, a member of Zilog's Z8051 Series of 8-bit microcontrollers.

Programming the Z51F3220 Device

The Z51F3220 MCU incorporates Flash memory to which a program can be written, erased, and overwritten while mounted on the Application Board. Additionally, Flash memory can be programmed or erased from within the user program.

Features

The Z51F3220 MCU Programmer offers the following features:

- Flash Size: 32KB
- Single power supply program and erase
- Command interface for fast program and erase operations
- Up to 10,000 Flash memory program/erase cycles at typical voltage and temperature
- Up to 100,000 Data EEPROM memory program/erase cycles at typical voltage and temperature
- Security features

Hardware Interface

The programming interface hardware consists of two signaling wires (DSDA and DSCL), power (V_{CC}), and Ground (GND).

Every byte provides a parity bit; the parity used is EVEN parity. After every byte, an acknowledgement bit is generated by the receiving device to indicate that it has received the byte; monitoring the bus for an acknowledgement is not required.

The DSDA is an open collector driver for both the device (IC) and the programmer (debugger). The DSCL is a tristated driver at the programmer. At different times during the programming process, interface signals can behave differently, as shown in Table 9.

Table 9. Signal Behavior During Programming

	Driver		Direction			
Signal	**Host**	**Device**	**Host During Byte Transfer**	**Device During During Byte Transfer**	**Host Between Bytes or Packets**	**Device Between Bytes or Packets**
DSDA	Open drain	Open drain	Input when Read command; output when Write command	Input when Write command; output when Read command	Output driven Low	Input
DSCL	Tristate	Tristate	Output	Output	Input; host monitors the line to check when the Slave stops driving it Low.	Output driven Low to indicate Busy status.

Pin Description

When used in conjunction with the On-Chip Debugger, the In-System Programmer uses Port 0 pins [1:0] as the Debugger Serial Clock Line (DSCL) and the Debugger Serial Data Line (DSDA).

Figure 1 shows the pin assignments for the 44-pin MQFP package. The red rectangle in the figure indicates the only pins used when programming the device.

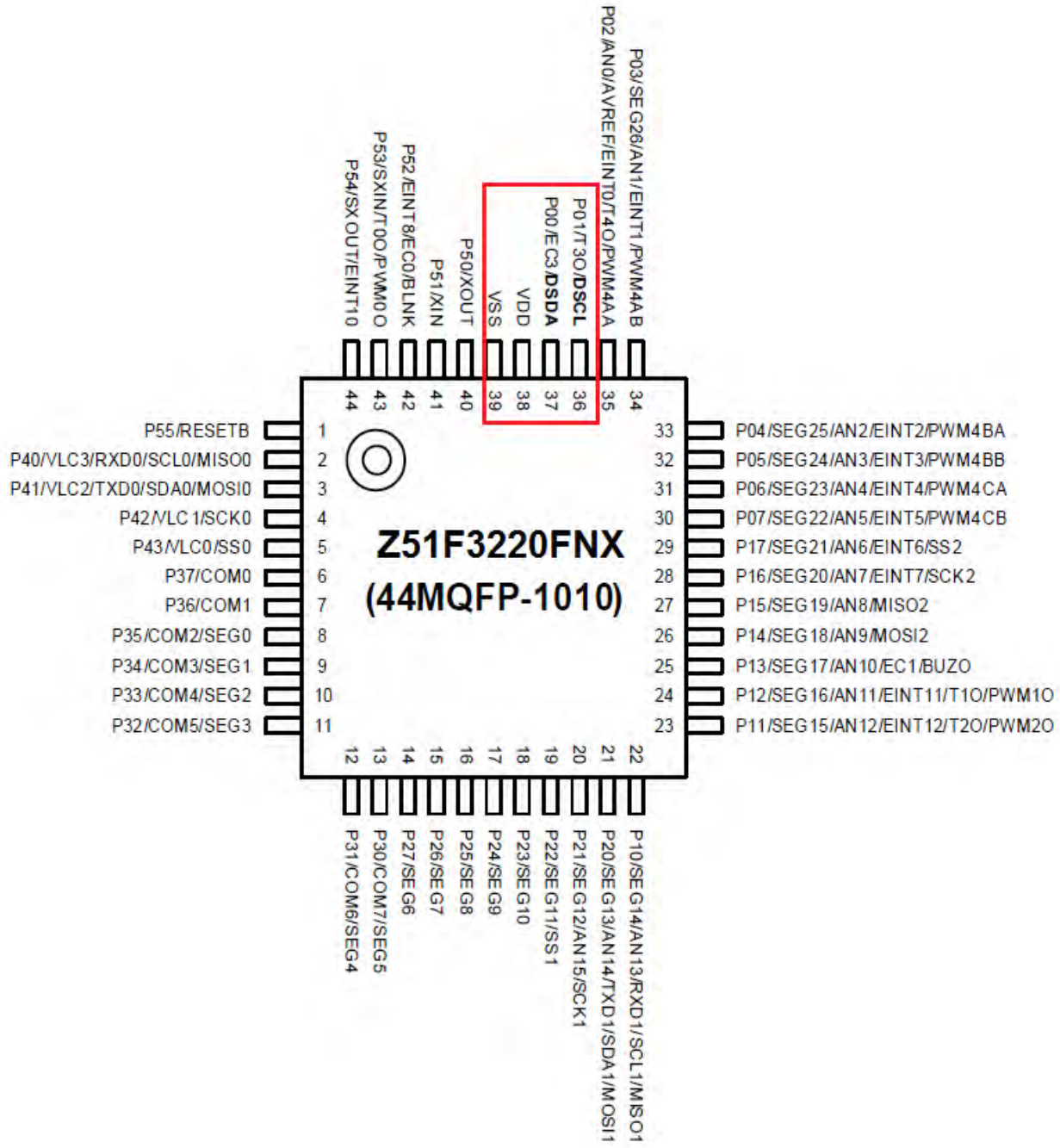

Figure 1. 44-Pin MQFP Pin Assignments

Figure 2 shows the pin assignments for the 32-pin SOP package. The red rectangles in the figure indicate the only pins used when programming the device.

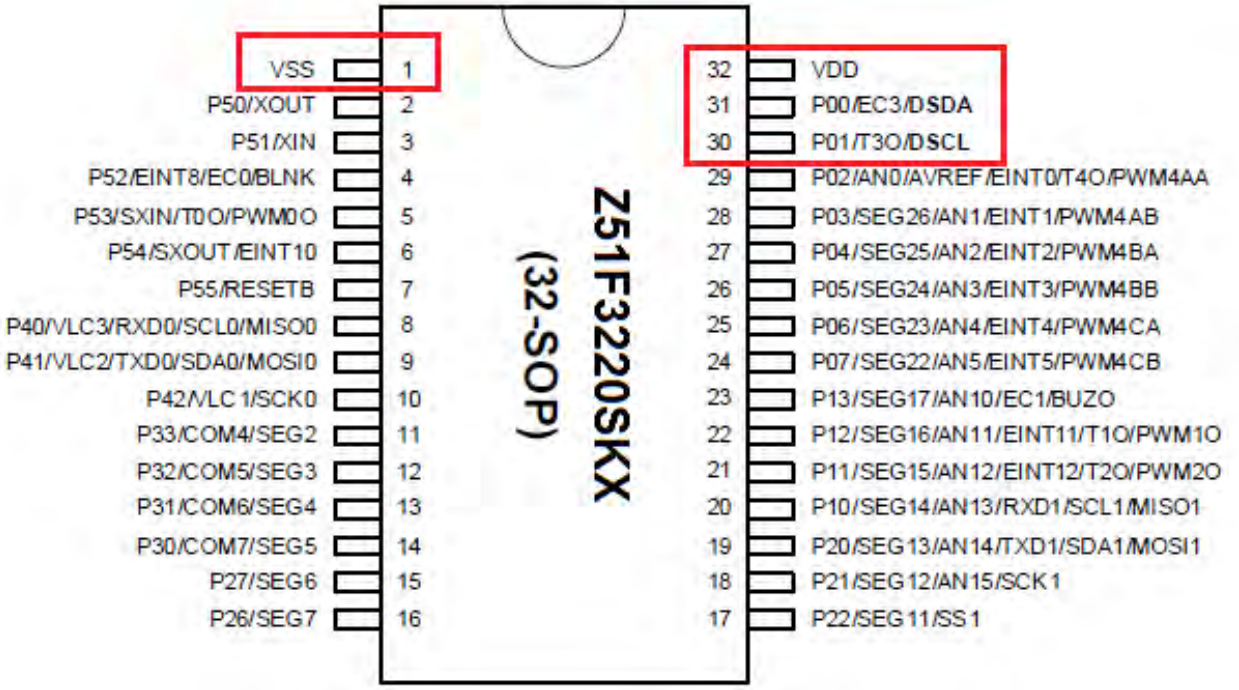

Figure 2. 32-Pin SOP Pin Assignments

Memory

Table 1 lists the Flash ROM, RAM and EEPROM sizes for each of the two Z51F3220 MCU package options.

Table 1. Memory Configurations

Device Name	Flash ROM	Package
Z51F3220FNX	32KB	44-pin MQFP
Z51F3220SKX		32-pin SOP

Program Memory

Figure 3 shows a map of the lower part of the program memory space.

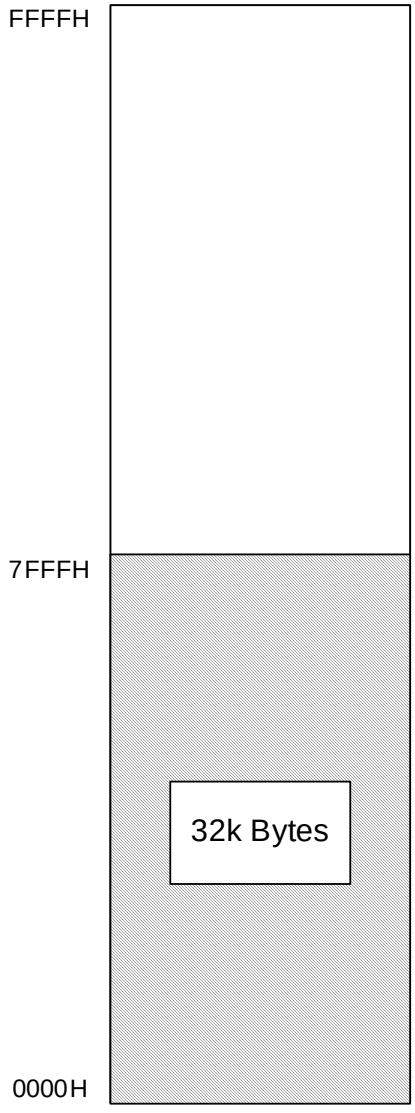

Figure 3. Program Memory

Programming Mode

This section describes how to program the Z51F3220 MCU with the Zilog On-Chip Debugger and the In-System Programmer, using Port 0 pins [1:0] as the DSCL and the DSDA.

Entering Programming Mode

After an initialization sequence, the Z51F3220 device will drive DSCL Low. When DSCL and DSDA are active Low, the programming interface is in an idle state between packets, as indicated by the circled regions in Figure 4.

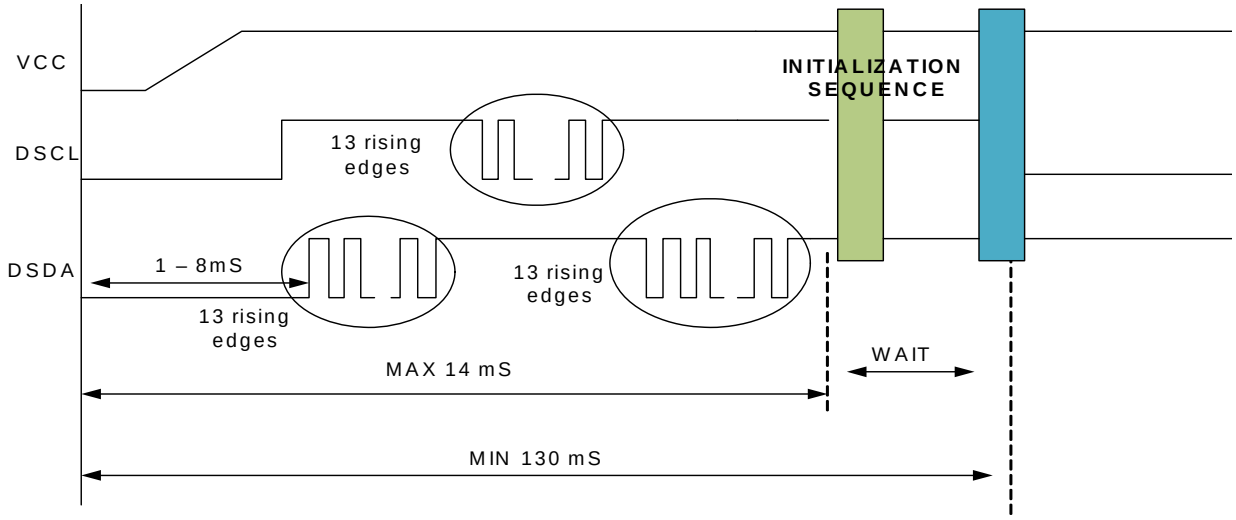

Figure 4. Entry Into Programming Mode

Data Interface Signalling

Each packet begins with a Start condition and ends with a Stop condition, as shown in Figure 5.

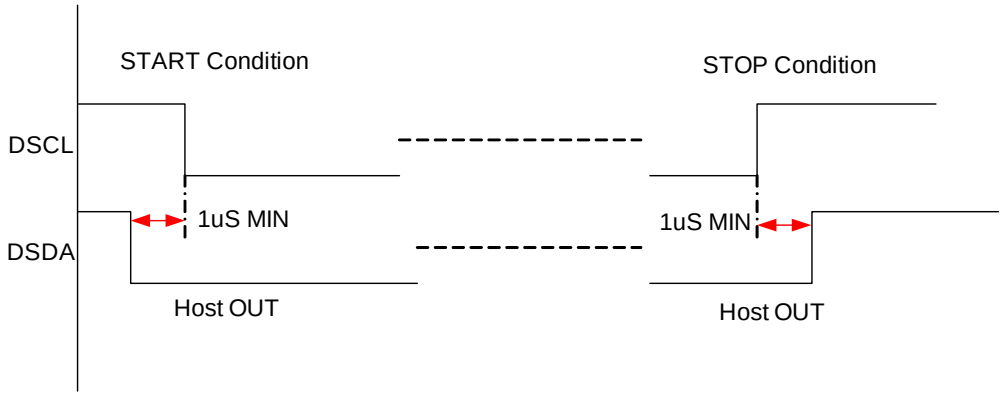

Figure 5. Data Interface Signal Timing

Host Write Operation

If the MCU is in a Busy state during a host write operation, the MCU pulls the DSCL line to GND; the host must wait for the line to be released, as indicated in Figure 6.

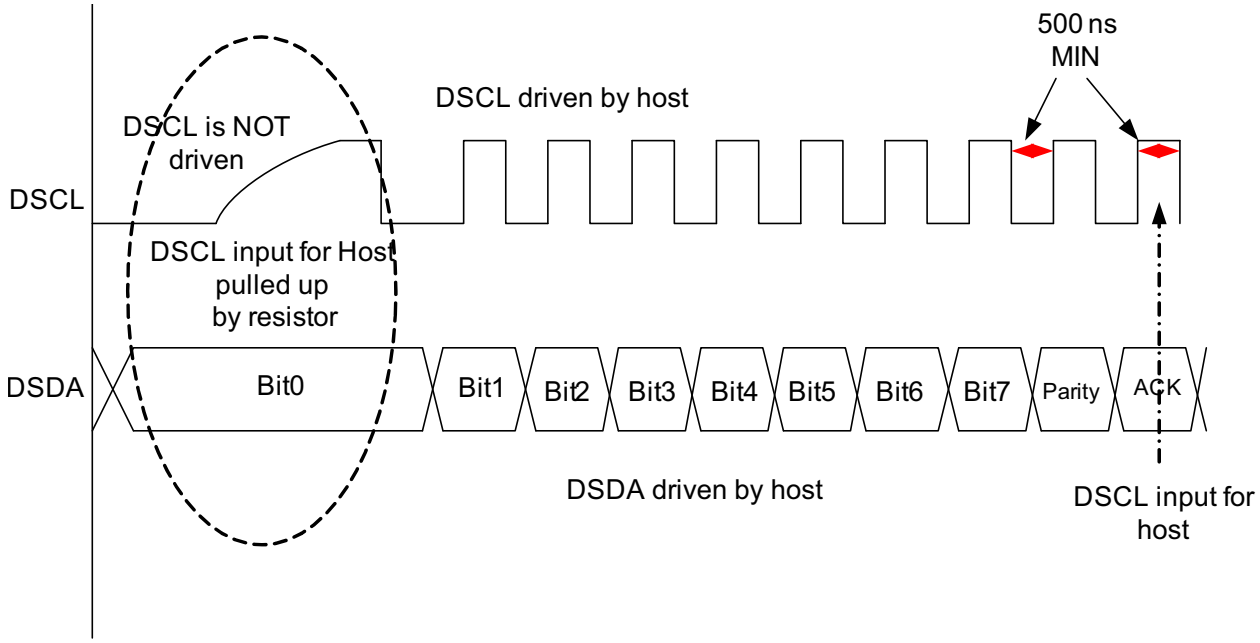

Figure 6. Host Write Timing

Note: In Figure 6, if DSDA is `0000_1100`, parity is 0; if DSDA is `0100_1100`, parity is 1. If communication occurs without an error, ACK = 0; otherwise, ACK = 1.

Host Read Operation

If the MCU is in a Busy state during a host read operation, the MCU pulls the DSCL line to GND; the host must wait for the line to be released, as indicated in Figure 7.

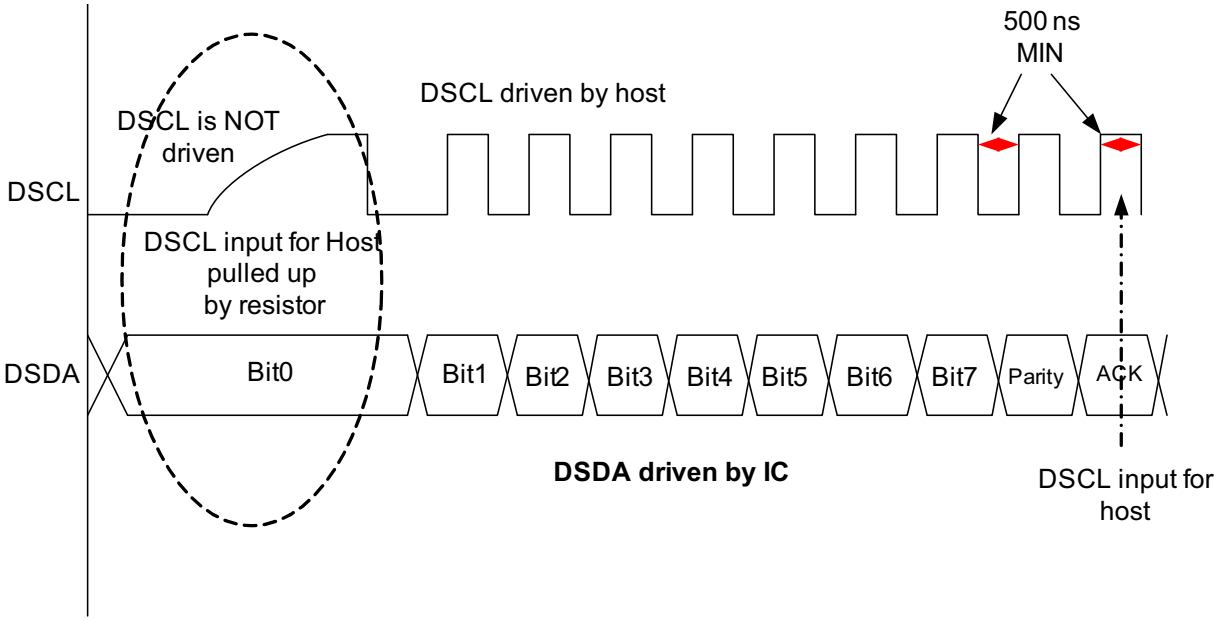

Figure 7. Host Read Timing

Note: In Figure 7, if DSDA is `0000_1100`, parity is 0; if DSDA is `0100_1100`, parity is 1. If communication occurs without an error, ACK = 0; otherwise, ACK = 1.

Command and Data Packets

Every packet consists of a command field, a target area field, an address field, and a data field. Table 2 indicates the Write Packet structure; Table 3 indicates the structure of the Read packet, and Figures 8 and 9 present the communication packet sequence and timing, respectively.

Table 2. Write Command/Data String

Start Condition	Write Command	Target Area	Addresses 7:0	Addresses 15:8	Addresses 24:16	Data Bytes	End Condition
	0x12	8 bits	8 bits	8 bits	8 bits	Data 0 .. DataN	

Table 3. Read Command String

Start Condition	Write Command	Target Area	Addresses 7:0	Addresses 15:8	Addresses 24:16	Data Bytes	End Condition
	0x22	Target Area	8 bits	8 bits	8 bits	Data 0 .. DataN	

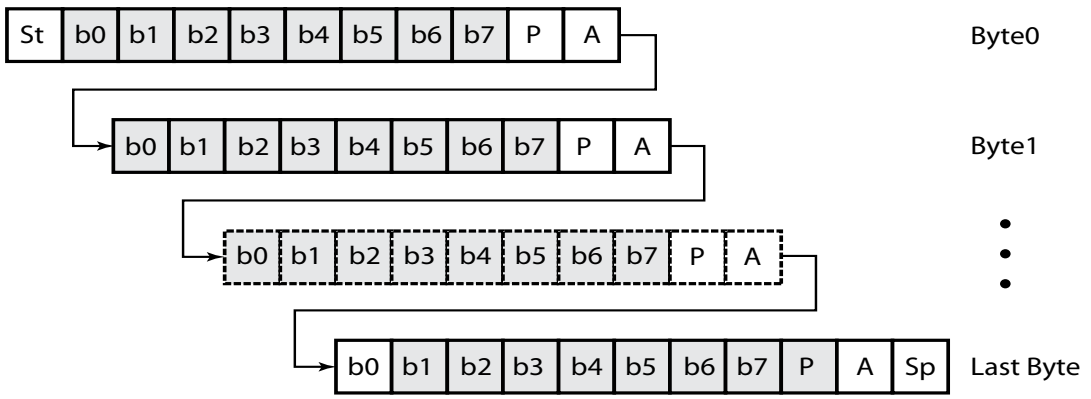

Figure 8. Communication Packet Sequence

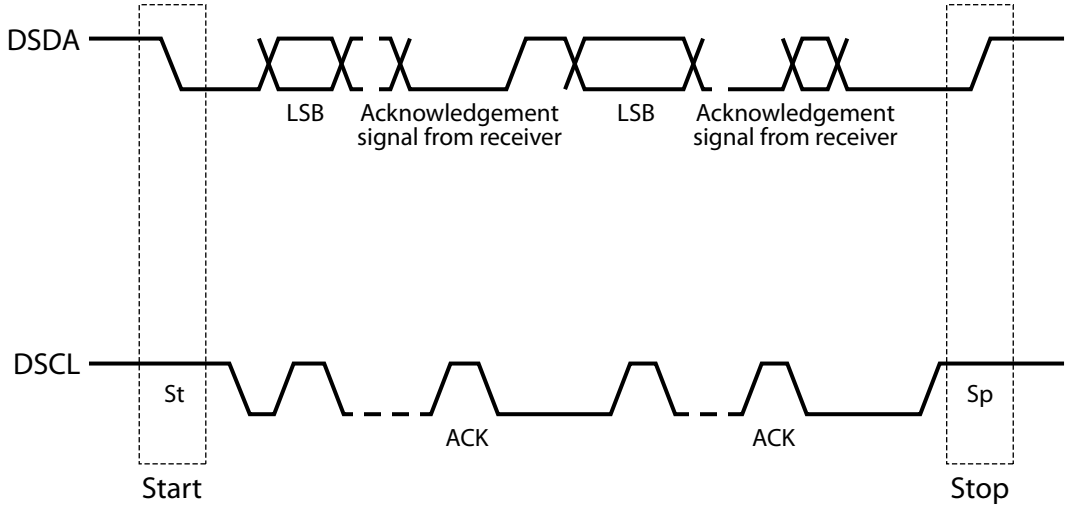

Figure 9. Communication Packet Timing

Target Areas

Table 4 lists the multiple target areas of the Z51F3220 MCU.

Table 4. Target Areas

Target Area Name	Function	Value
OCD_CODE	Z51F3220, Code Area	0x10
OCD_XDATA	Z51F3220, OCD Data Area in the range 0x8000–0x803F	0x11
OCD_SFR	Z51F3220, Special Function Registers	0x13
OCD_BDC	Z51F3220, Background Debugging Controller	0x20
OCD_DBG	Z51F3220, Debugging Logic	0x21
OCD_TRG	Z51F3220, Debugging Trigger	0x22

Flash Programming

This section describes the Z51F3220 MCU's Flash architecture and controller registers plus the operations required for successful Flash programming and debugging.

Flash Architecture

Figure 10 depicts the relationship between the page buffer and Flash memory.

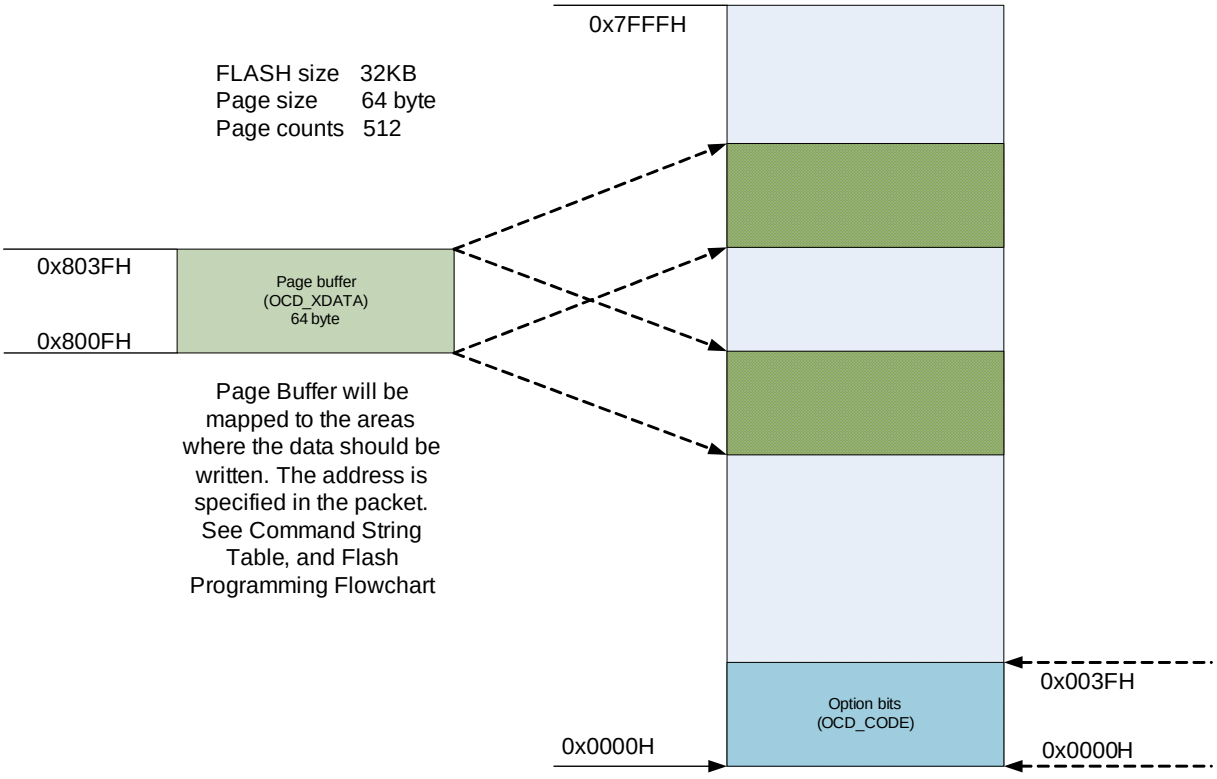

Figure 10. The Z51F3220 MCU Flash Memory Architecture

Flash Controller Registers

The Flash Controller registers, listed in Table 5, are located in the OCD_SFR Area.

Table 5. Flash Controller Register Map

Name	Address	Bit	Function	Notes
Regdata	SFR (F9h)	[7]	V_{PP} Out Enable (VPPOE)	
		[2]	Memory Target (Flash: 0; Option: 1)	
		[0]	ERASE Mode (AE: 0: Page Erase; 1: Bulk Erase)	
FSADRH	SFR (FAh)	[3:0]	Flash Address [19:16]	
FSADRM	SFR (FBh)	[7:0]	Flash Address [15:8]	
FSADRL	SFR (FCh)	[7:0]	Flash Address [7:0]	
FIDIR	SFR (FDh)	[7:0]	FMCR Operation Enable (0xA5: Enable)	
FMCR	SFR (FEh)	[7]	Flash Busy Flag (0: Done; 1: Busy)	
		[2:0]	**Flash Register**	
			000: Wait	
			001: Page Buffer Reset	
			010: Erase	FIDR 0xA5
			011: Program	FIDR 0xA5
			100: Hard Lock	FIDR 0xA5
Page Buffer	XDATA (8000h–803Fh)		Flash Page Buffer	

Flash Operation

Table 6 lists the many Flash operations that can be performed when programming the Z51F3220 MCU. It is important to note that the page buffer will be cleared by writing 0x01 to the FMCR Register. Option reading is enabled even if the Z51F3220 device is locked.

Table 6. Flash Operation

Operation: Step and Register Contain	Option Page Erase	Flash Page Erase (One Page)	Bulk Erase	Option Page Program	Flash Page Program	Hard Lock
Page Buffer Clear	FMCR = 0x01	FMCR= 0x01	X	FMCR = 0x01	FMCR = 0x01	X
Page Buffer Write	Fill page buffer with 0x00	Fill page buffer with 0x00	Fill page buffer with 0x00	X		
REGDATA	0x04	0x00	0x01 (Flash) 0x05 (Flash + options)			
FSADRH	X	0x00	X	0x00	0x00	X
FSADRM	X	0x00	X	0x00	0x00	X
FSADRL	X	0x40 (1 page)	X	0x40 (1 page)	0x40 (1 page)	X
FIDR	0xA5	0xA5	0xA5	0xA5	0xA5	0xA5
FMCR	0x02	0x02	0x02	0x03	0x03	0x04

Option Data

Table 7 describes each of the Z51F3220 MCU's Flash option bits.

Table 7. Flash Option Data Register

Address	Bit	Description
0x3E	[7:3] TRIMDATA	**Reserved** Trim Data values are programmed at the factory and must not be modified. Read and save these values before erasing, then write these values during programming.
	[2] PAEN	**Protection Area Enable** 0 : Disable. 1: Enable.
	[1] PASS	**Protected Area Size Selection** 00 : 3.8KBytes(0x0100 ~ 0x0FFF) 01 : 1.7KBytes(0x0100 ~ 0x07FF) 10 : 768 Bytes(0x0100 ~ 0x03FF) 11 : 256 Bytes(0x0100 ~ 0x01FF)

Table 7. Flash Option Data Register (Continued)

Address	Bit	Description
0x3F	[7] R_P	**Read Protection** 0 : Disable. 1: Enable.
	[6] HL	**Hard Lock Enable** 0: Disable. 1: Enable.
	[5:1]	**Reserved** These bits are reserved and must be programmed to 00000.
	[0] RSTS	**RESETB Select** 0 : I/ O pin. 1 : RESETB pin.

Entering Flash Programming Mode

From the moment device is powered up, it takes about 210 ms for it to enter Programming Mode. The following procedure must be performed during this period.

1. Apply power to device while holding DSCL and DSDA low.
2. After power is OK release DSCL, and start counting time.
3. Wait for 8 mS and generate 13 rising edges on DSDA.
4. Keep DSDA High and generate 13 rising edges on DSCL.
5. Keep DSCL High and generate 13 rising edges on DSDA. DSDA and DSCL should be High at the end.
6. To initialize device generate a following packet[1]:

```
Start
0x12
0x20
0x04
0x00
0x00
0x80
End

Start
0x12
0x21
```

1. There is no need to monitor for an ACK during these packets. If an ACK is being generated by the device, simply follow with the packet.

```
0x00
0x00
0x00
0x80
End

Start
0x12
0x21
0x00
0x00
0x00
0x90
End
```

a. Wait until the time counter reaches approximately 220ms to stop driving the DSCL line (i.e., place the driver into a tristate mode) and start monitoring the DSCL line. The device will drive the line Low to indicate a Busy status. Wait until the device stops driving the DSCL line Low to generate a Start condition.

Figure 11 presents a waveform of the DSCL and DSDA signals.

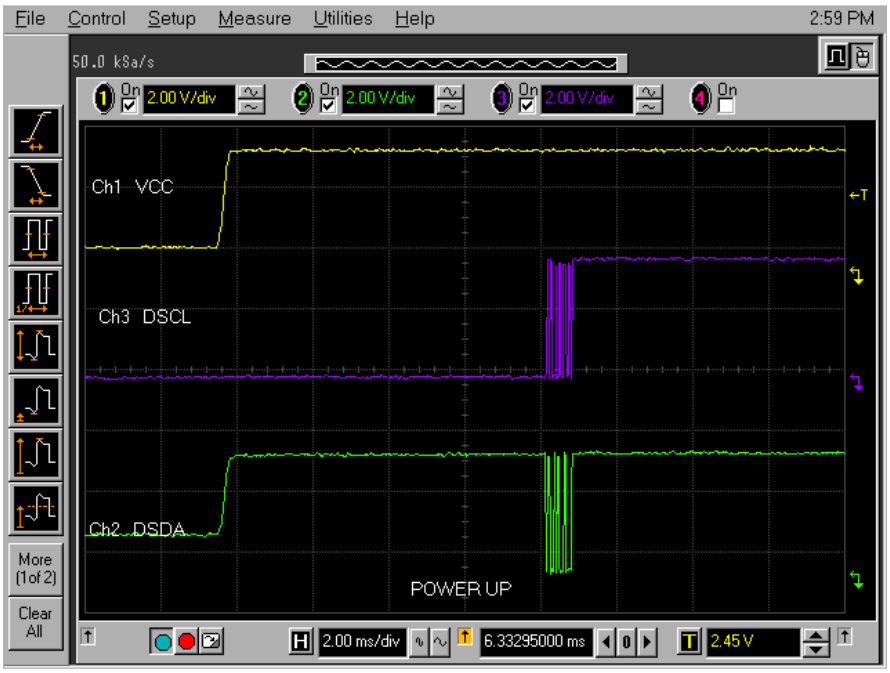

Figure 11. Initializing and Entering Flash Programming Mode

Entering Debug Mode

To reset the Z51F3220 device and place it into Debug Mode, observe the following procedure:

1. Reset the device by sending the following packet; ensure that both the DSDA and DSCL lines remain Low.

```
Start
0x12
0x20
0x00
0x00
0x00
0x00
End
```

2. Send the following sequence to the device to enter Debug Mode:

```
Start
0x12
0x21
0x00
0x00
0x00
0x90
End
```

With the device now operating in Debug Mode, you can begin programming; refer to the Flash programming flowcharts shown in Figures 12 and 13.

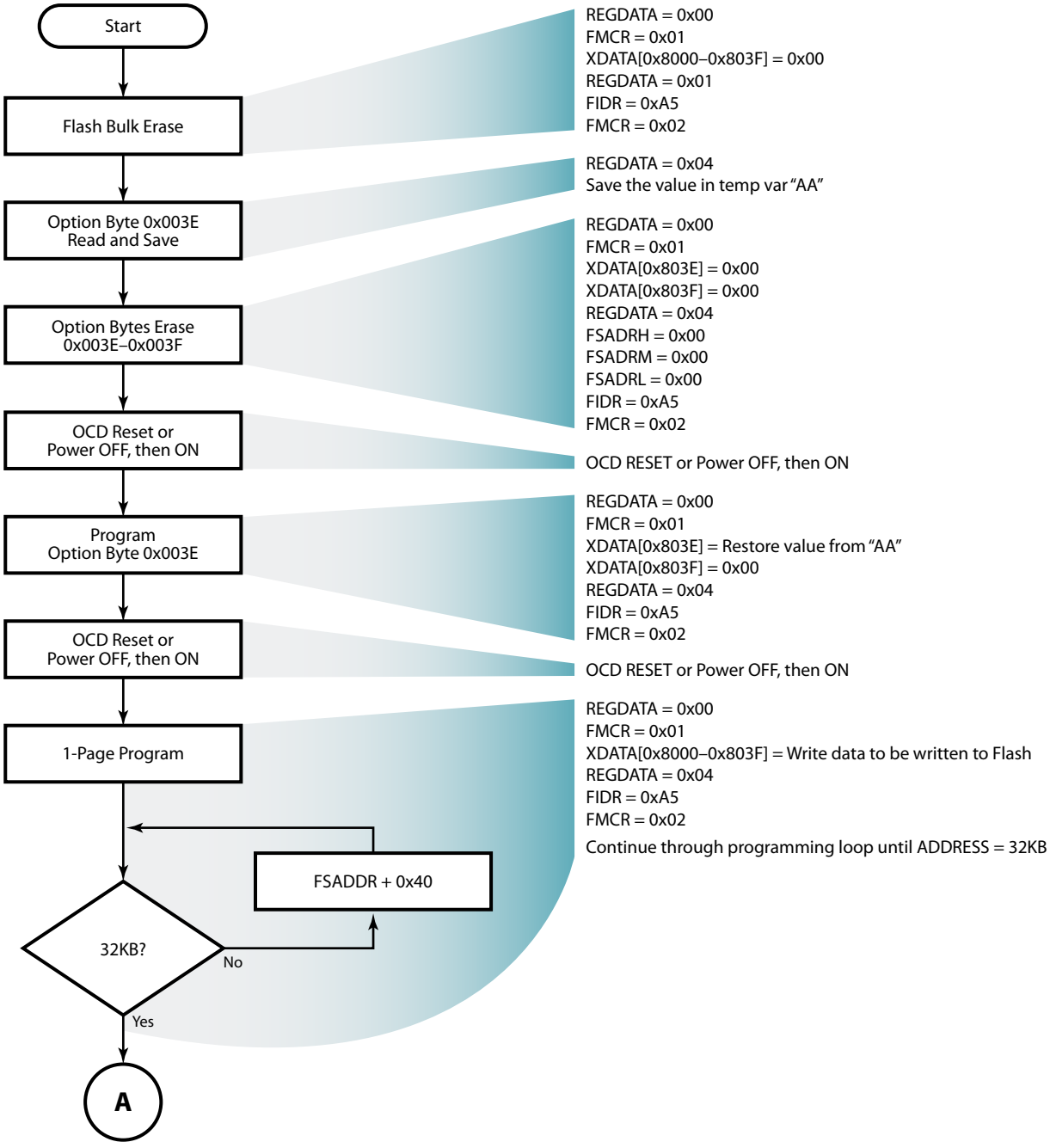

Figure 12. Flash Programming Flow, #1 of 2

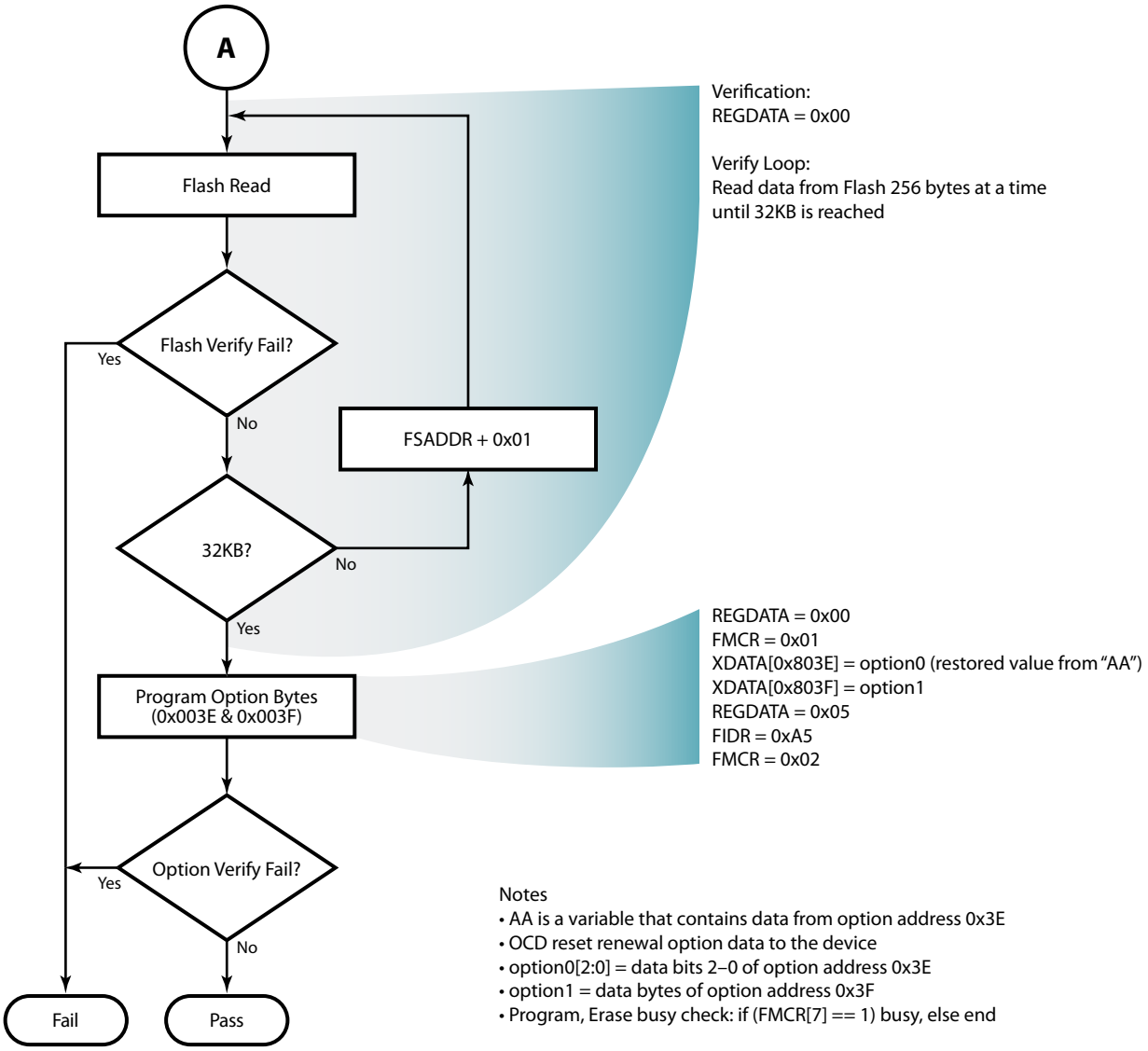

Figure 13. Flash Programming Flow, #2 of 2

Packet and Timing Structure

In Figure 14, the first three bytes after the power-up sequence are:

- 0x12: WRITE command
- 0x20: OCD_BDC, Area 20 Background Debug Controller

- 0x04: Address, LSB

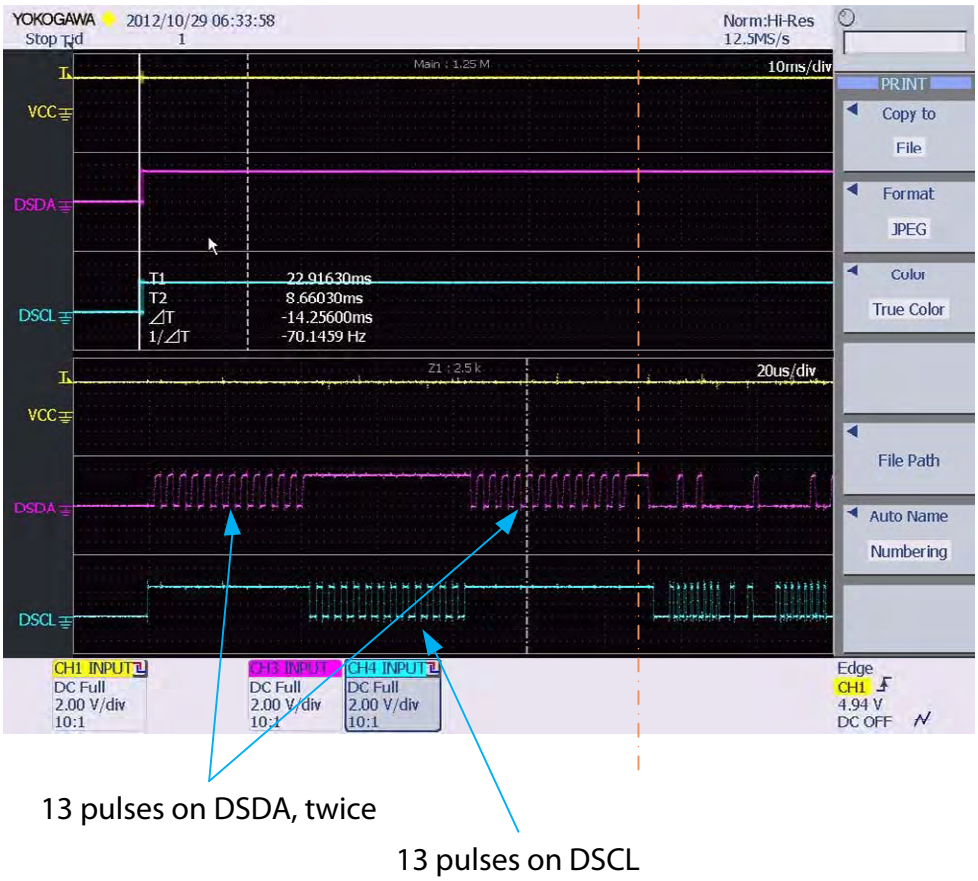

Figure 14. Packet and Timing Structure