Design Guide: TIDM-1011

Tamagawa T-Format Absolute-Encoder Master Interface Reference Design for C2000 MCUs

Description

C2000™ microcontroller (MCU) Position Manager technology offers an integrated solution to interface to the most popular digital- and analog-position sensors, which eliminates the necessity for external field-programmable gate arrays (FPGAs) or application-specific integrated circuits (ASICs). The Position Manager BoosterPack™ is a flexible, cost-effective platform intended for evaluating various encoder interfaces and is designed to work with multiple C2000 MCU LaunchPad™ development kits. The software of this reference design specifically targets implementation of the T-Format, which is a digital, bidirectional interface for position encoders. The highly optimized and easy-to-use software reference implementation and examples included in this reference design enable T-Format, position-encoder operation using the Position Manager BoosterPack.

Features

- Flexible, low-voltage, BoosterPack evaluation platform for position-encoder interfaces
- Integrated MCU solution for T-Format encoder interface without additional FPGA requirements
- Easy interface T-Format commands through driver functions and data structure provided by interface function
- Support for unpacking received data and optimized cyclic redundancy check (CRC) algorithm
- Supports a clock frequency of 2.5 MBPS and verified operation up to 100-m cable length
- Includes evaluation software example showcasing the T-Format implementation

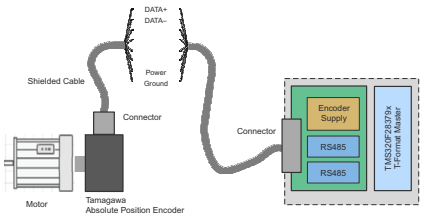
Applications

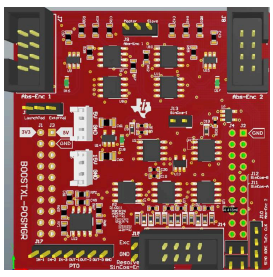
- Industrial
- Motor Drives

Resources

TIDM-1011	Design Folder
LAUNCHXL-F28P65X LAUNCHXL-F28379D LAUNCHXL-F280039C LAUNCHXL-F280049C LAUNCHXL-F280025C	Tools Folder
SN65HVD78, TLV702, TPS22918-Q1	Product Folder
C2000WARE-MOTORCONTROL-SDK	Tools Folder

Ask our TI E2E™ support experts





1 System Description

Industrial drives, like servo drives, require accurate, highly-reliable, and low-latency position feedback. The T-Format protocol, from Tamagawa, is designed for serial transfer of digital data between a sensor and a controller. The sensor can be an encoder (linear, rotary, or angle), a touch probe, or an accelerometer. The subsequent electronics, or controller, includes numerical controls, servo amplifiers, and programmable-logic controllers.

The TIDM-1011 design implements a T-Format encoder interface to a C2000 LaunchPad. T-Format is a pure-serial, digital interface, based on the RS-485 standard. T-Format is capable of transmitting position values, along with other physical quantities, and allows reading and writing of the internal memory of the encoder. The transmitted-data types include absolute position, turns, temperature, parameters, and diagnostics. Commands transmitted to the encoder from the interface select the response-data types.

Figure 1-1 shows a T-format encoder connected to a BOOSTXL_POSMGR plus F28379D LaunchPad encoder interface.

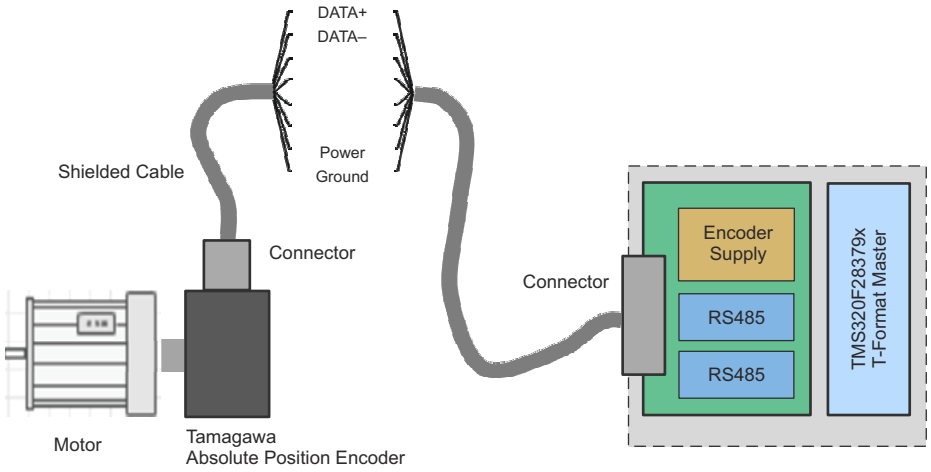


Figure 1-1. Industrial Servo Drive With T-Format Position Encoder Interface

The position encoder with T-Format connects to the TIDM-1011 device through a single, 4-wire, shielded cable. RS-485 is used as the physical layer for T-Format encoders. The four wires used are:

- DATA+ and DATA- : differential signals for communication data
- Power and Ground: encoder power supply and ground

The Texas Instruments C2000 T-Format (PM_tformat) encoder interface implementation enables interfacing a T-Format encoder to a C2000 device without external hardware such as an FPGA or CPLD. The reference implementation features:

- 2.5 MBPS clock frequency as required by the T-Format protocol
- Integrated cable-propagation delay compensation to enable variable cable length, verified up to 100 m
- Software driver functions:
  - Perform a transaction with the encoder. This consists of sending a request and receiving the response.
  - Calculate a CRC
  - Compare CRC received with a calculated CRC
  - Packing and unpacking the data

This reference implementation includes all the source code. Any changes needed for the implementation can be made by users as needed by their application.

Note

Only the basic interface drivers for the commands defined in the T-Format specification are provided. All the higher-level application software must be developed by users using the basic interface provided by this implementation.

1.1 Key System Specifications

Table 1-1. Key System Specifications

PARAMETER	SPECIFICATIONS	DETAILS
Input voltage	5 V(1)	Section 3.3.1
Output voltage (encoder)	5 V	Section 3.3.1
Protocol supported	T-Format	Tamagawa
Frequency (encoder interface)	Approximately 2.5 MBPS	Tamagawa
Encoder bits	T-Format protocol standard	Tamagawa
CPU cycles	C2000 T-Format encoder interface benchmarks	Section 3.3.5

(1) The time of the encoder connected to the TIDM-1011 device determines the current limit of this supply. TI recommends a generic, bench-top, adjustable, power supply with an adjustable current limit.

2 System Overview

The C2000 T-Format TIDM-1011 reference design is a combination of hardware and software. The core hardware components are a C2000 real-time microcontroller (MCU) and a RS-485 transceiver. The C2000 LaunchPad and the TIDM-1011 boosterPack, which contains the RS-485 transceiver, are the boards used in this implementation. The C2000 Motor Control SDK package contains the necessary software. This software includes a library, which implements key T-format interface features, along with a system-level example to demonstrate T-Format communication.

The T-Format encoder interface leverages the C2000 CLB (Configurable Logic Block) and the SPI (Serial Peripheral Interface) modules. The CLB controls the SPI clock and compensates for cable propagation delay. The CLB also controls the RS-485 transceiver transmit enable. The SPI module acts as the send/receive interface to the RS-485 physical layer. The firmware, which is written in C, runs on the C28x of the C2000 MCU.

The C2000 LaunchPad can provide power for the TIDM-1011 RS-485 transceiver and 5V for the encoder. 5V can also be supplied separately if the encoder specifications require a higher current than the LaunchPad can provide.

During start-up, the application running on the C28x initializes the MCU clocks and configures the pin-mux. The MCU's SPI and CLB are also configured as required to send and receive data.

2.1 Block Diagram

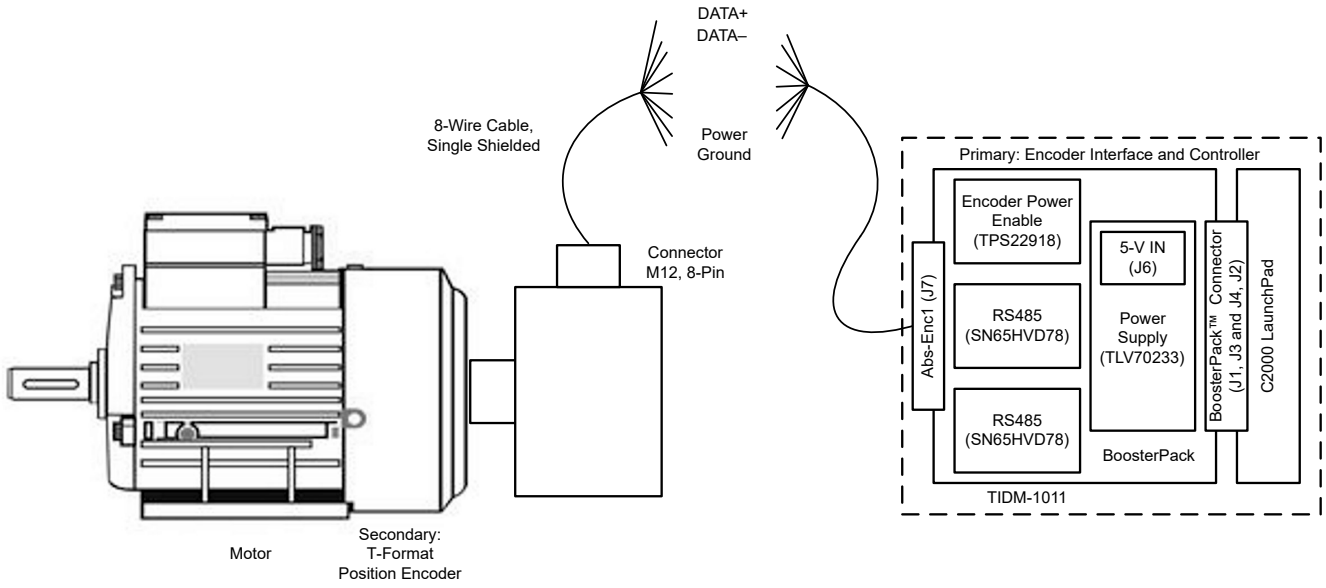


Figure 2-1. TIDM-1011 System Block Diagram

2.2 Highlighted Products

The TIDM-1011 reference design hardware consists of a C2000 LaunchPad plus a BOOSTXL-POSMGR BoosterPack. This section covers the key devices used. For more information on each of these devices, see their respective product folders at TI.com.

2.2.1 C2000 Real-Time MCU LaunchPad

Multiple LaunchPad kits support the TIDM-1011 reference design (refer to Table 2-1). Each of the C2000 Real-Time Microcontrollers listed in Table 2-1 feature the Configurable Logic Block (CLB). The encoder interface makes extensive use of the CLB. The CLB peripheral is exclusive to C2000 devices and allows users to incorporate custom logic without the need for an external FPGA or CLPD. The CLB is composed of submodules that combine together to enable custom digital logic. Submodules include: Finite State Machines (FSM), Lookup Tables (LUT), and counters. The CLB also interfaces with existing on-chip control peripherals to enhance functionality and provide design options.

To learn more about the CLB, visit the C2000 Academy Configurable Logic Block module.

Devices with the CLB include:

- TMS320F28379D MCU:

  Provides 800 MIPS of total system performance between dual, 200-MHz, C28x CPUs and dual, 200-MHz, real-time-control coprocessors (CLA). This powerful MCU contains 1MB of on-board flash and includes highly-differentiated peripherals, such as 16-bit or 12-bit analog-to-digital converters (ADCs), comparators, 12-bit digital-to-analog converters (DACs), delta-sigma sync filters, HRPWMs, eCAPs, eQEPs, CANs, and more. Find the full device features and specifications at the TMS320F28379D device product folder.
- TMS320F280039C MCU:

  Provides 240 MIPS between a 120 MHz C28x CPU and 120 MHz CLA. This MCU contains up to 384 kB of on-chip flash and includes 3 12-bit ADCs, enhanced Configurable Logic Blocks (CLB), and more. Find the full device features and specifications at the TMS320F280039C device product folder.
- Table 2-1 lists other supported devices, their product folders, and their LaunchPad Development Kits.

Table 2-1. Supported Devices and LaunchPads

LaunchPad Development Kit	MCU Device Product Folder (2)
LAUNCHXL-F28379D	TMS320F28379D
LAUNCHXL-F280049C	TMS320F280049C
LAUNCHXL-F280025C	TMS320F280025C
LAUNCHXL-F280039C	TMS320F280039C
Not Available (1)	TMS320F28388D
LAUNCHXL-F28P65X	TMDS320F28P650DK9

(1) The TMS320F28388D device family does not have a LaunchPad development kit. You must supply the connections to an RS-485 physical interface through another means. Options include (1) your own hardware, (2) a controlCard with wires to the BOOSTXL_POSMGR, or (3) the TMXIDDK379D.

(2) The TIDM-1011 reference design requires a C2000 LaunchPad with an MCU featuring the Configurable Logic Block (CLB) type 1 or later. Devices supported at the time of this release are shown. Additional devices may be available.

2.2.2 SN65HVD78

The SN65HVD78 device combines a differential driver and a differential receiver, which operate from a single, 3.3-V power supply. The differential outputs of the driver and the differential inputs of the receiver are internally connected to form a bus port suitable for half-duplex (two-wire bus) communication. These devices feature a wide, common-mode voltage range, which makes the devices suitable for multipoint applications over long cable runs.

Find the full device features and specifications at the SN65HVD78 product folder.

2.2.3 TLV702

The TLV702 series of low-dropout (LDO) linear regulators are low-quiescent current devices with excellent line and load-transient performance. All device versions have thermal shutdown and current limit for safety. The devices regulate to specified accuracy with no output load.

Find the full device features and specifications at the TLV702 product folder.

2.2.4 TPS22918-Q1

The TPS22918-Q1 is a single-channel load switch, with configurable rise time and configurable quick-output discharge. The device contains an N-channel MOSFET that can support a maximum-continuous current of 2 A. The switch is controlled by an on and off input, which can interface directly with low-voltage control signals.

Find the full device features and specifications at the TPS22918-Q1 product folder.

2.3 Design Considerations

This section provides:

1. Overview of the Tamagawa T-Format protocol.
2. Overview of the C2000 T-Format encoder interface.
3. TIDM-1011 hardware (BOOSTXL-POSMGR BoosterPack) implementation.
4. C2000 MCU implementation, including the required input/output, CRC calculations, and the CLB design.
5. Overview of the C2000 T-Format encoder interface software library.

Note

This section provides implementation details only. For information related to:

- Hardware requirements, setup and testing: Refer to Section 3.
- Software: Installing and running the software: Refer to: "C2000 T-Format Encoder Interface Software Guide" (html, pdf). The software guide includes documentation for:
  - Communication demonstration
  - T-Format application programmer interface (API)
  - Incorporating the library into your own solution
  - Migration from previous versions

2.3.1 Tamagawa T-Format Protocol

Tamagawa is a manufacturer of encoder technology used for obtaining high-precision position information in machine tools, robotics, motor drives and so forth. Tamagawa rotary encoders consist broadly of two types: incremental or absolute. Incremental encoders provide a train of pulses, while the absolute-type provides absolute digital values. Absolute encoders include both single-turn and multi-turn types.

The TIDM-1011 reference design focuses on an absolute-type which provides a digital output through an RS-485 line driver. The protocol format of the transaction supported by TIDM-1011 is known as T-Format.

Note

This section provides an overview of the T-Format protocol. For specific information, refer to the T-format specification available from Tamagawa.

Note

Not all encoders produced by Tamagawa use the T-Format protocol. Check the specification of your encoder.

The C2000 T-Format encoder interface provides the required physical layer for a controller to communicate with an encoder. This encoder interface provides the RS-485 drive control to send and receive digital information with the encoder. In this context, a T-Format transaction is the transmission of a request from the controller plus the response back from the encoder. From the perspective of the encoder interface, a transaction can be divided into FRAME_STATES shown in Figure 2-2:

1. IDLE: No RS-485 activity
2. TRANSMIT_DATA: The controller sends a request to the encoder
3. WAIT_FOR_START: Wait for the encoder's response
4. RECEIVE_DATA: The controller receives the encoder's response
5. Back to IDLE.

This pattern is repeated for each transaction with the encoder.

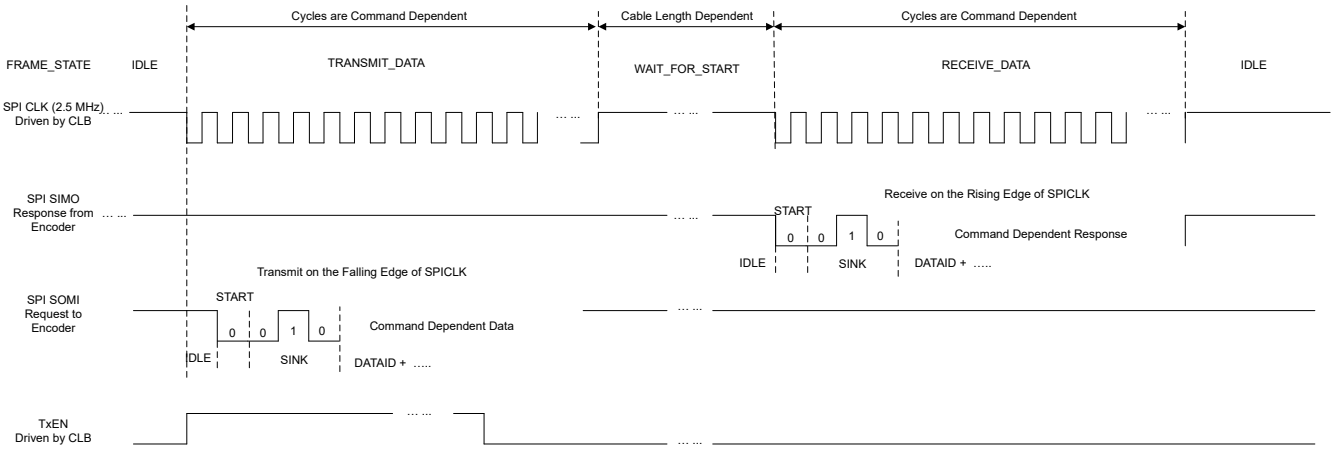


Figure 2-2. T-Format Frame

Note

Figure 2-2 includes information related to the C2000 encoder interface implementation. For example, the CLB drives the SPI CLOCK and the TxEN signal.

The T-format communication protocol is broadly classified into three types of transactions: data readout, reset, and EEPROM access. Each transaction has a unique Data ID defined by the protocol. The Data ID is used to identify the specific request made by the controller through the encoder interface.

Table 2-2. T-Format Transactions

Transaction Type	Data ID	Transaction
Readout	ID 0	Absolute data in one revolution
	ID 1	Multi-turn data
	ID 2	Encoder ID
	ID 3	All of the above plus the encoder error status
Reset	ID 7	Reset absolute data in one revolution
	ID 8	Reset multi-turn data
	ID C	Reset errors
EEPROM	ID D	Read encoder's EEPROM
	ID 6	Write to encoders EEPROM

Each transaction consists of 10-bit fields. Each field has the format shown in Table 2-3. The first bit is a start bit (always 0) and the last bit is a delimiter bit (always 1). The content of the 8-bits of data between the start bit and delimiter depend on the specific type of field.

Table 2-3. T-Format Field Format

	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7	Bit 8	Bit 9	Bit 10
Content	Start bit Always 0	8 data bits: Least significant bit first. The content depends on the specific field. These 8 bits of data are included in the CRC calculation.								Delimiter bit Always 1

The fields defined by the T-Format protocol are:

ControlField (CF)	The first field in every request and every response. The ControlField includes the unique Data ID for the transaction.
StatusField (SF)	Status information from the encoder.
DataFields (DFx)	Information from the encoder. The content and number of DataFields depend on the transaction. Examples of DataFields include the encoder's ID, position information, and error codes. Up to 8 DataFields are possible.
CRCField	An 8-bit Cyclic Redundancy Check (CRC) of the data. A CRCField is the last field of an EEPROM read or write request. The CRCField is always the last field in the encoder's response.
EEPROM AddressField (ADF)	Address to be read or written in an EEPROM transaction.
EEPROM DataField (EDF)	Contains the data read, or the data to be written, in an EEPROM transaction.

Note

For the specific contents of each field, refer to the T-Format specification available from Tamagawa.

The fields used in a request are shown in Table 2-4. To begin a transaction, the controller sends a request through the encoder interface. The request starts with the ControlField which includes the Data ID. The encoder uses the Data ID to identify the exact transaction requested. For a readout or reset request, only the ControlField is required. In the case of an EEPROM read or write the controller also sends an EEPROM AddressField and an EEPROM DataField (for a write) followed by a CRCField.

Table 2-4. T-Format Request Fields

Request type	Fields Transmitted			
Readout	ControlField			
Reset	ControlField			
EEPROM Write	ControlField	EEPROM AddressField	EEPROM DataField	CRCField(1)
EEPROM Read	ControlField	EEPROM AddressField	CRCField	

(1) The CRC calculation includes the 8-bits of data in the Control, EEPROM Address and EEPROM Data fields. The start bits and delimiter bits are excluded.

The fields used in an encoder's response depend on the specific request. For readout and reset transactions (Table 2-5), the encoder responds with an echo of the ControlField, follwed by a StatusField and one or more DataFields. Lastly the encoder always sends a CRCField. The CRCField can be used as an integrity check of the received data.

Table 2-5. T-Format Response Fields for Readout and Reset

Data ID	Type	Control Field(1)	Status Field	DataFields (DF0:DF1:....DF7) + CRC (2) (3)								
ID 0	Read	CF	SF	ABS0(4)	ABS1	ABS2	CRC					
ID 1	Read	CF	SF	ABM0(5)	ABM1	ABM2	CRC					
ID 2	Read	CF	SF	ENID(6)	CRC							
ID 3	Read	CF	SF	ABS0	ABS1	ABS2	ENID	ABM0	ABM1	ABM2	ALMC(7)	CRC
ID 7	Reset	CF	SF	ABS0	ABS1	ABS2	CRC					
ID 8	Reset	CF	SF	ABS0	ABS1	ABS2	CRC					
ID C	Reset	CF	SF	ABS0	ABS1	ABS2	CRC					

(1) CF: ControlField. Matches the ControlField sent in the request.
(2) DF: DataField. Up to 8 fields depending on the transaction.
(3) The CRCField is always transmitted immediately after the last used DataField. The CRC includes the 8-bits of data in the CF + SF + DataFields used. The start-bit and delimiter of each field is excluded.
(4) ABS: Absolute data in one revolution. Uses 3 fields.
(5) ABM: Multi-turn data. Uses 3 fields.
(6) ENID: Encoder ID. One field.
(7) ALMC: Encoder error. One field

In the following data readout example, the controller requests the multi-turn data (Data ID 1). Referencing Table 2-5, the response DataFields correspond to the multi-turn data (ABM0:ABM1:ABM2).

Table 2-6. Data Readout Example

Request:	ControlField for Data ID 1					
Response:	ControlField for Data ID 1	StatusField	DataField0	DataField1	DataField2	CRC

For a EEPROM transaction, the encoder responds with the ControlField + EEPROM AddressField + EEPROM DataField + CRCField as shown in Table 2-7.

Table 2-7. EEPROM Read/Write Response Fields

Data ID	Request	Field 0	Field 1	Field 2	Field 3
ID 6	Write	ControlField	EEPROM AddressField	EEPROM DataField (1)	CRCField(3)
ID D	Read	ControlField	EEPROM AddressField	EEPROM DataField (2)	CRCField

(1) Data read from the encoder's EEPROM.
(2) Data written to the EEPROM. This is an echo of the DataField in the request.
(3) The CRC calculation includes the 8-bits of data in the Control, EEPROM Address and EEPROM Data fields. The start bit and delimiter of each field are excluded.

2.3.2 C2000 T-Format Encoder Interface Overview

Communication over a T-Format encoder interface is primarily achieved by the following components:

- CPU (C28x)
  - Configures the device, CLB, and SPI
  - Packs and unpacks data
  - Calculates the transmit CRC for EEPROM commands
  - F2837xD only: Calculates CRC for received data
  - Compares calculated CRC with received CRC
- Configurable logic block (CLB)
  - Controls the SPI clock
  - Controls the transmit enable signal to the RS-485 transceiver
  - Measures, and compensates for, cable propagation delay as required by the interface
  - Calculates the CRC of the received data (feature not available on F2837xD)
- Serial peripheral interface (SPI)
  - Performs the encoder-data transmit and receive
- Device interconnects (XBARs or CLB XBARs)
  - Route signals into and out of the CLB and the device
- External interface block
  - TIDM-1011 board with RS-485 differential line driver

Note

Depending on the device features, different methods are used to calculate CRCs. Refer to Section 2.3.5 for information.

Figure 2-3 shows the T-Format encoder interface connections.

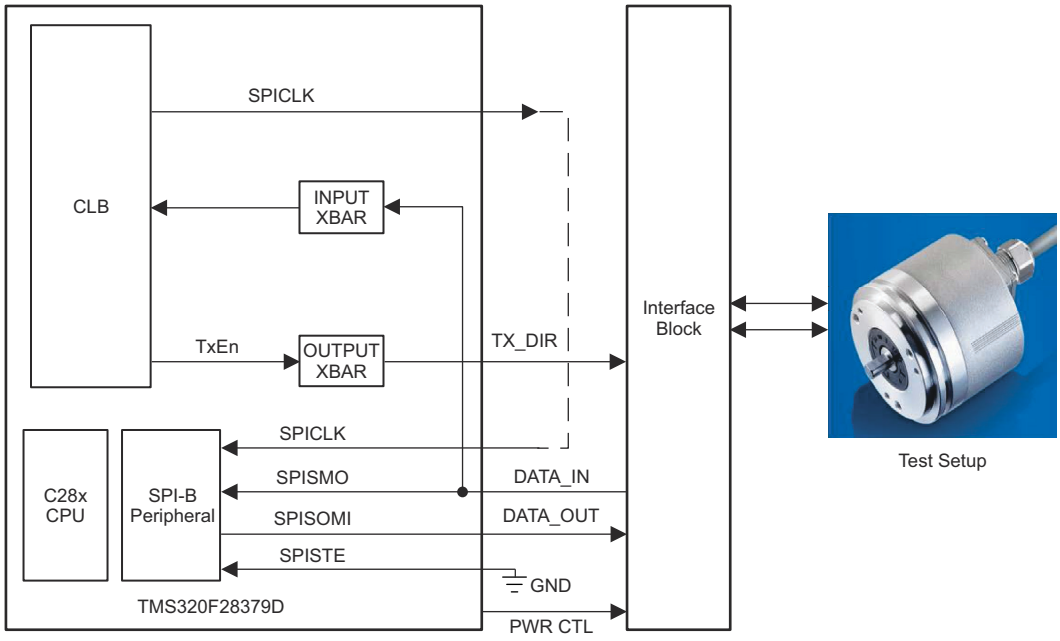


Figure 2-3. T-Format Implementation Diagram Inside TMS320F28379D

The remainder of this section describes the following aspects of the design:

- The TIDM-1011 hardware (BOOSTXL-POSMGR)
- C2000 MCU resources including the CLB design
- C2000 software used by the encoder interface

2.3.3 TIDM-1011 Board Implementation

The TIDM-1011 board provides the following:

- Differential line driver and receiver for RS-485 communication between a C2000 MCU and the encoder.
- TxEN signal routed from the MCU to the direction control of the RS-485 driver/receiver.
- SPICLK signal routed to a GPIO where it can be controlled by the CLB peripheral. This connection is optional for all supported devices except the F2837x family.

Note

The TIDM-1011 daughter card is identical to the Position Manager BoosterPack plug-in module (BOOSTXL-POSMGR), which means the TIDM-1011 can interface with several position-encoder types. The board is fully populated by default. This reference design focuses on the T-Format absolute encoder protocol, and the hardware blocks not used can be ignored.

Table 2-8 lists the connectors used by the TIDM-1011 T-Format implementation and their functions.

Table 2-8. TIDM-1011 Board and BOOSTXL-POSMGR Connectors

CONNECTOR	DESCRIPTION	USED BY TIDM-1011
Abs-Enc-1 (J7)	T-Format and other absolute encoders	Yes, LaunchPad Site 2
Abs-Enc-2 (J8)	T-Format and other absolute encoders	No
Abs-Enc-2 Breakout (J10)	Allows two absolute encoders at site two using jumpers	No
SinCos (J14)	SinCos encoder	No
Resolver (J14 and J15)	Resolver interface with 15-V excitation circuitry	No
PTO (J17)	Pulse-train output	No
J1, J3 and J4, J2	BoosterPack connector	Yes
J6	5-V DC supply input	Yes

Table 2-8. TIDM-1011 Board and BOOSTXL-POSMGR Connectors (continued)

CONNECTOR	DESCRIPTION	USED BY TIDM-1011
J16	15-V DC resolver excitation input	No

Figure 2-4 shows the encoder support on each site of the LaunchPad.

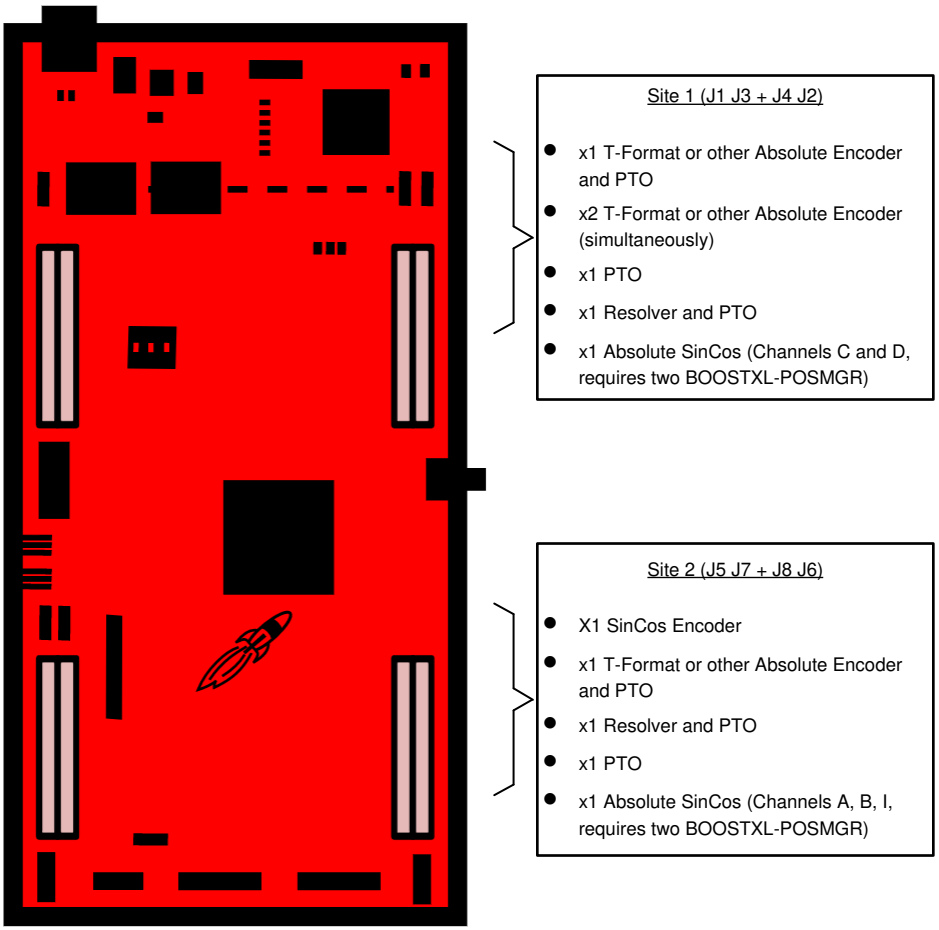


Figure 2-4. TIDM-1011 Board and BOOSTXL-POSMGR Encoder Support

As provided, TIDM-1011 uses LaunchPad Site 2 and BOOSTXL-POSMGR's Encoder 1 connections. Figure 2-5 shows the connections. The complete schematic of the TIDM-1011 BoosterPack can be downloaded from the BOOSTXL-POSMGR product page.

Note

The F2837xD device requires an external connection between the CLB generated clock (CLB_SPI_CLK) and the SPICLK pin. On all other devices the CLB can directly drive SPICLK and an external connection is not required.

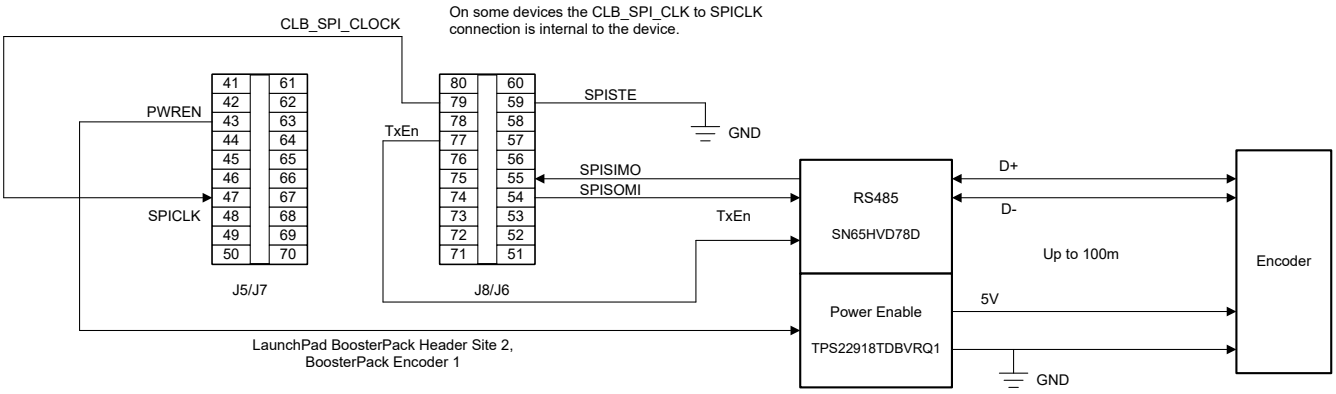


Figure 2-5. LaunchPad Site2 with BOOSTXL-POSMGR Encoder 1 Pinout

2.3.4 MCU Resource Requirements

Table 2-9 lists the C2000 Real-Time MCU resources used by the TIDM-1011 reference design. Specifics for each device are described in Section 2.3.5.

Table 2-9. TIDM-1011 Resource Usage

RESOURCE NAME and Quantity	TYPE	PURPOSE
CLB x 2	1 Tile	Provides the SPI clock, delay compensation, and TxEn control. If the tile is changed, then routing in/out of the CLB must also be updated.
	1 Tile (Optional)	Calculate the response CRC as the message is received. This option is only available on devices with CLB Type 2 or later.
GPIO x 2	I/O	• CLB output, RS-485 direction control (TxEN) • CPU encoder power control (PwrCtl)
GPIO x 1	I/O (F2837xD only)	• CLB output of CLB_SPI_CLK • CLB Type 1: route this pin externally to the SPICLK input. • CLB Type 2, or later: clock the SPI module directly from the CLB. An external connection is not required, but can be useful for test and debug.
INPUTXBAR or CLB_INPUTXBAR x 1	Module, I/O	Connect the SPI SIMO pin to the CLB.
OUTPUTXBAR or CLB_OUTPUTXBAR x 1	Module, I/O	Connects the CLB to TxEN (direction control).
SPI x 1	Module and I/Os	One SPI instance to transmit and receive on the RS-485 physical layer. The SPI clock is controlled by the CLB.
CPU and Memory	Module	CPU and memory use for various functions.

2.3.5 Device-Specific Resource Usage

Device-specific resources used by TIDM-1011 include:

- Resource to perform the CRC calculations.
- Input and output signals and the specific CLB tile instance(s) used.

2.3.5.1 CRC Calculations

Cyclic Redundancy Check (CRC) is an error detection mechanism used in communication networks and data storage. The device resources available on a C2000 MCU to calculate a CRC have increased over time. TIDM-1011 uses different resources depending on device features and whether the data is transmitted or received. The resource usage is summarized in Table 2-10.

Table 2-10. Resource Used for CRC Calculations

Device	Receive Data CRC	Transmit Data CRC
F2837xD	C28x plus lookup table	C28x plus lookup table
F28004x	Configurable Logic Block	C28x lookup table
All others	Configurable Logic Block	VCRC extension to C28x

- **C28x Lookup Table** is available on all C28x devices. This method, however, is the slowest and requries RAM memory for look-up table storage. TIDM-1011 only uses a look-up table on devices without another capability.
- **Configurable Logic Block (CRC)** is available on devices with CLB type 2, or later. The CLB calculates a CRC using a counter configured as a Linear Feedback Shift Register (LFSR). This method has been used to calculate the CRC of the encoder response as the data is being received. This frees C28x bandwidth as no additional calculations are required. The C28x reads the CRC result directly from the counter register. The cost of this method is CLB tile resources and code to configure the tile. The CLB CRC implementation is documented in Section 2.3.7.
- **VCRC** is a C28x instruction set extension specifically for CRC calculations. The implementation is faster than the C28x lookup-table for longer messages. In addition, the VCRC does not require RAM space to store a lookup-table. For devices with the VCRC module, this method has been used to calculate the transmit data CRC used in EEPROM read/write transactions.

Note

The CRC method can be selected in the T-Format library header file.

2.3.5.2 Input, Output Signals and CLB Tiles

This section describes the input/output and CLB tile connections used on each device.

Note

In the input/output diagrams, a letter in a colored circle indicates an off-page connection.

- Communication tile: Connectors A, B, C, and G are described in Section 2.3.6
- CRC tile: Connectors B, F, E, and A are described in Section 2.3.7

The GPIO pins and SPI module used depend on the device-specific LaunchPad and BOOSTXL-POSMGR BoosterPack pinout. The connections into, and out of, the CLB depend on the features of that device. The specific tile instances used depend on the capability of the tile to override other signals such as the SPICLK.Table 2-11 summarizes the input/output resources used by each device family. The I/O figure for each device details which GPIOs and CLB tiles were used.

Table 2-11. Input / Output and Tile Summary per Device

Device	I/O Figure	CLB RX CRC Tile (3)	SPI Module	CLB to SPICLK	Other I/O
F2837xD	Figure 2-6	No	SPI-B	Externally connected	Tile 4 override of EPWM4B. (1) Device INPUTXBAR and OUTPUTXBAR
F28004x	Figure 2-7	Yes	SPI-B	Driven directly by CLB	Device INPUTXBAR and OUTPUTXBAR
F28003x F28002x	Figure 2-8	Yes	SPI-B	Driven directly by CLB	CLB_INPUTXBAR and CLB_OUTPUTXBAR
F2838x	Figure 2-9 (2)	Yes	SPI-B	Driven directly by CLB	CLB_INPUTXBAR and CLB_OUTPUTXBAR
F28P65x	Figure 2-10	Yes	SPI-D	Driven directly by CLB	CLB_INPUTXBAR and CLB_OUTPUTXBAR

(1) CLB tile 4 overrides the EPWM4B output signal to control GPIO7. No other ePWM functionality is used.
(2) The F2838x family is not supported by the LaunchPad development platform. The pins used by the TMDXIDDKF273XD hardware platform are shown in the I/O diagram.

(3) For devices with a CRC tile, connections between the communication tile and CRC tile are shown in the I/O diagrams.

Note

At the time this document was published, the F2837xD and F28004x are supported by TMDXIDDKF273XD development kit projects. For convenience, the GPIOs for both the BOOSTXL_POSMGR and TMDXIDDKF273XD platform have been provided in the I/O diagrams.

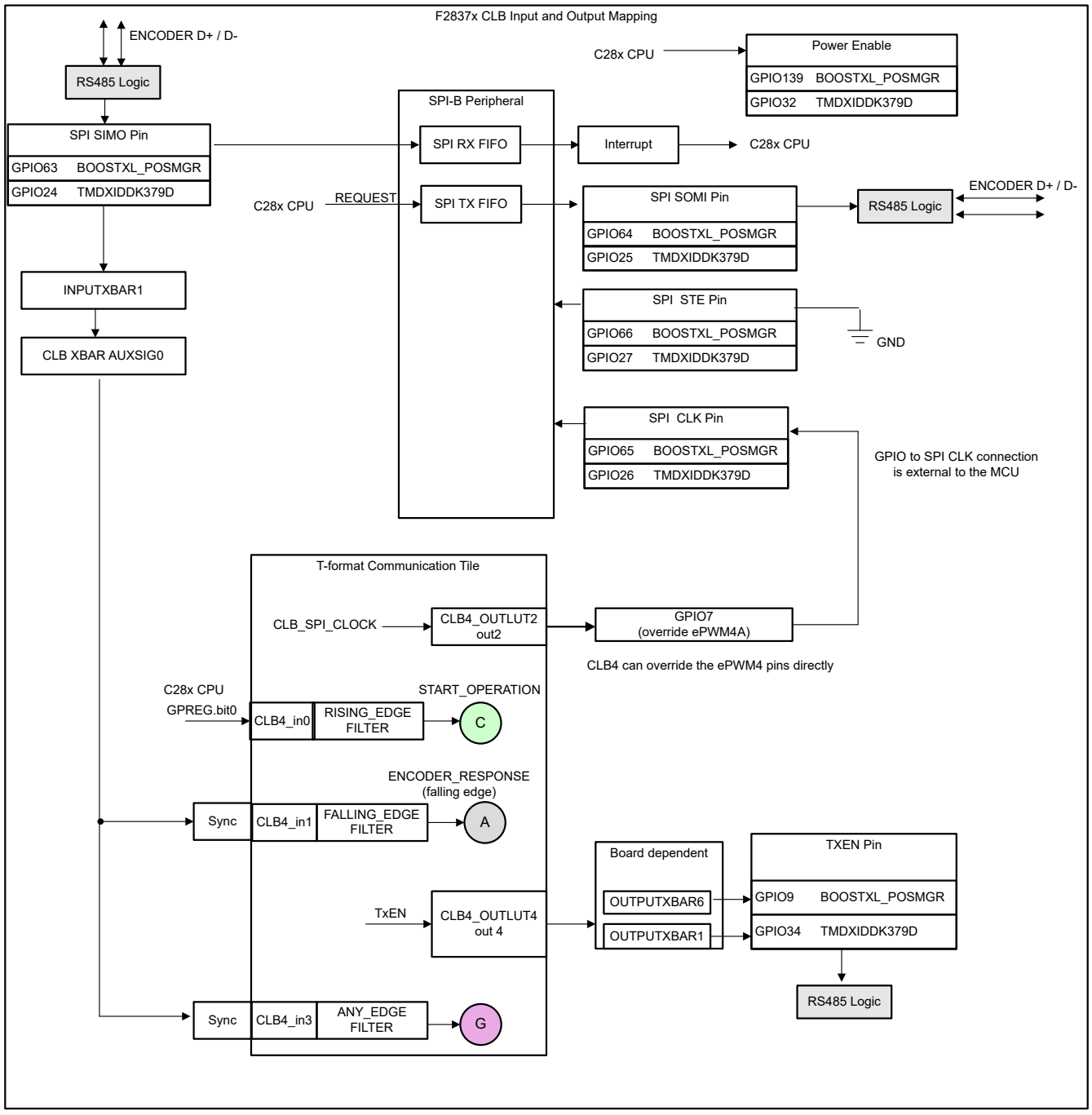


Figure 2-6. F2837xD Input, Output, and CLB Usage for BOOSTXL_POSMGR and TMDXIDDKF273XD

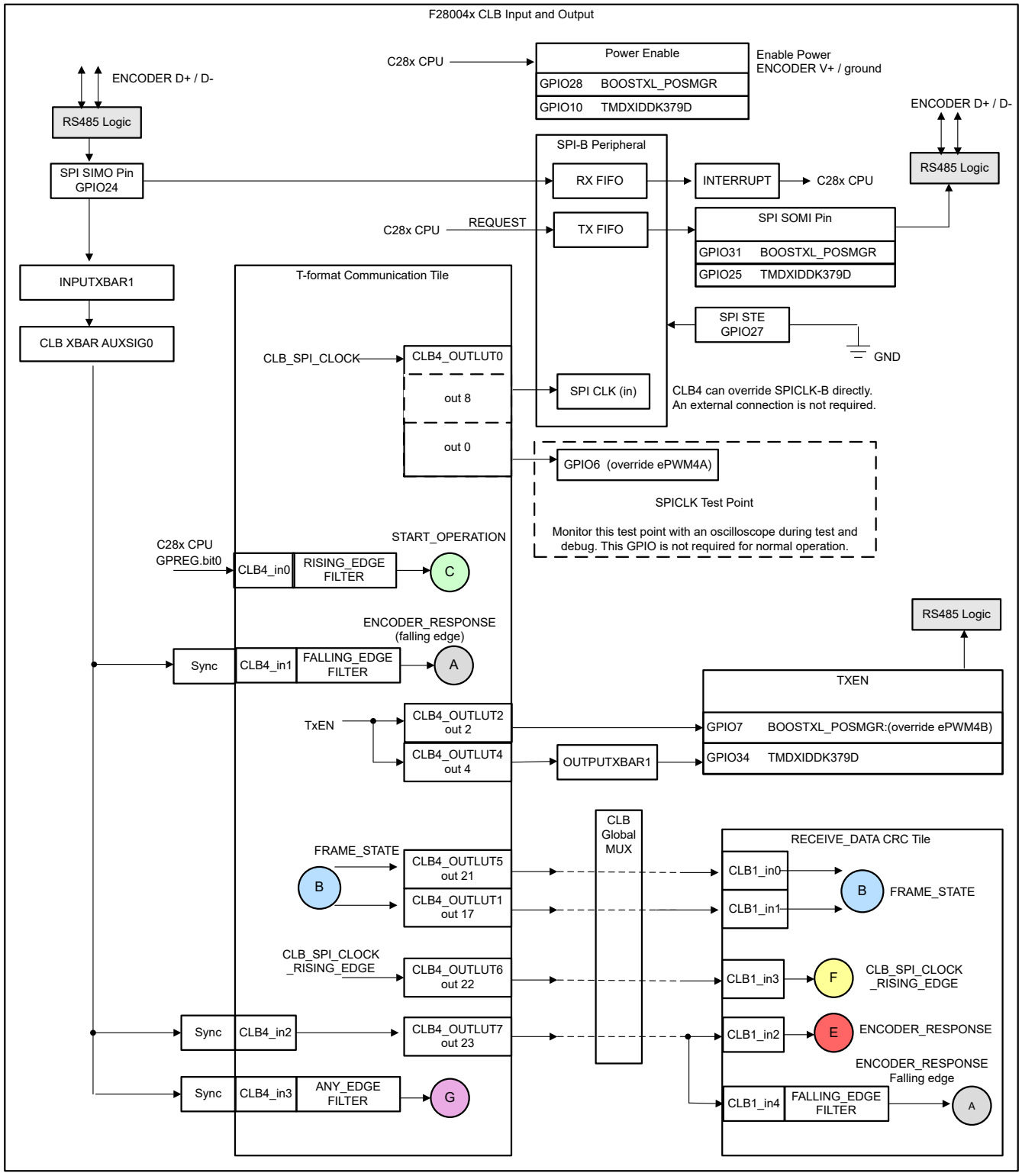


Figure 2-7. F28004x Input, Output, and CLB Usage for BOOSTXL_POSMGR and TMDXIDDKF273XD

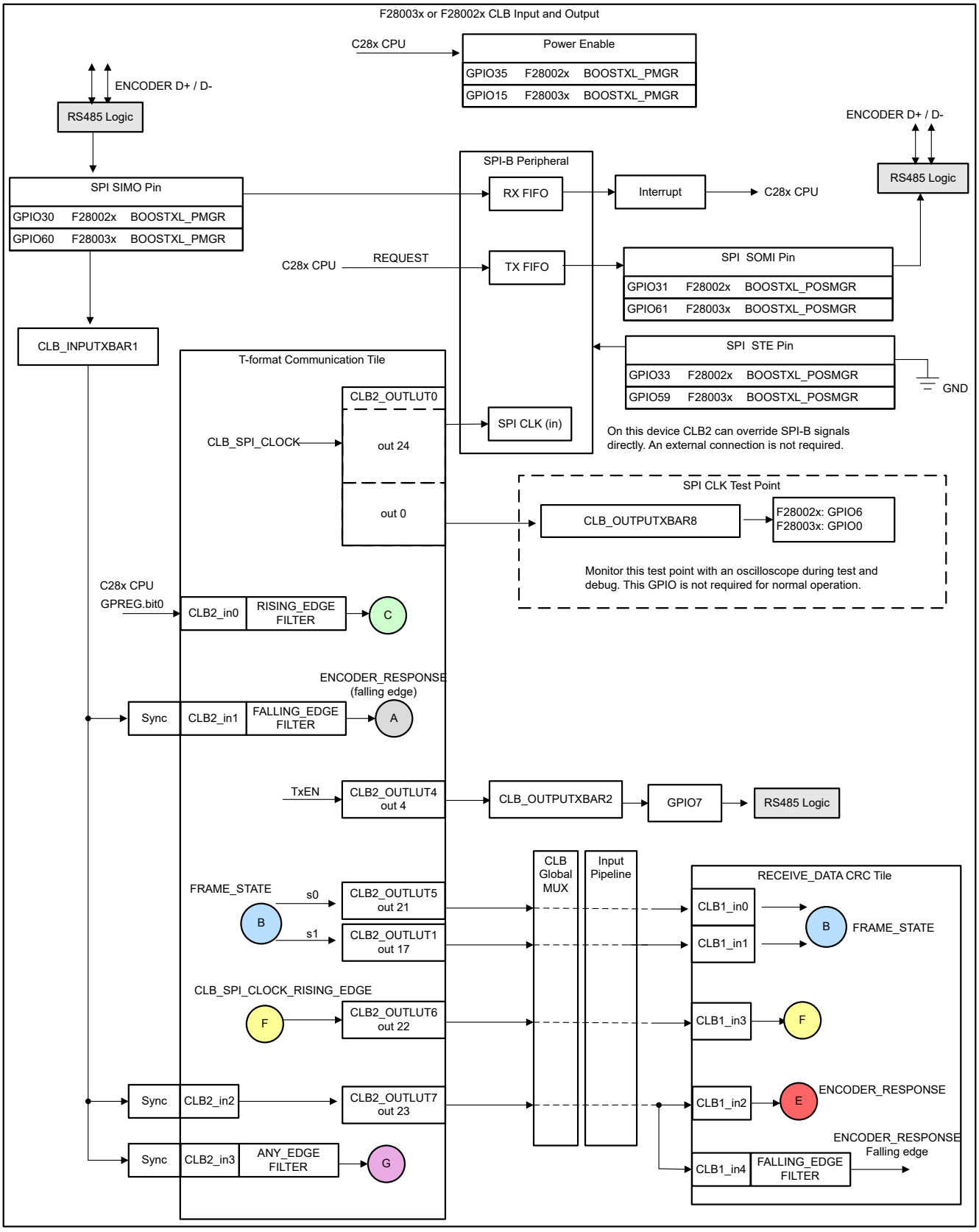


Figure 2-8. F28002x and F28003x Input, Output, and CLB Usage for BOOSTXL_POSMGR

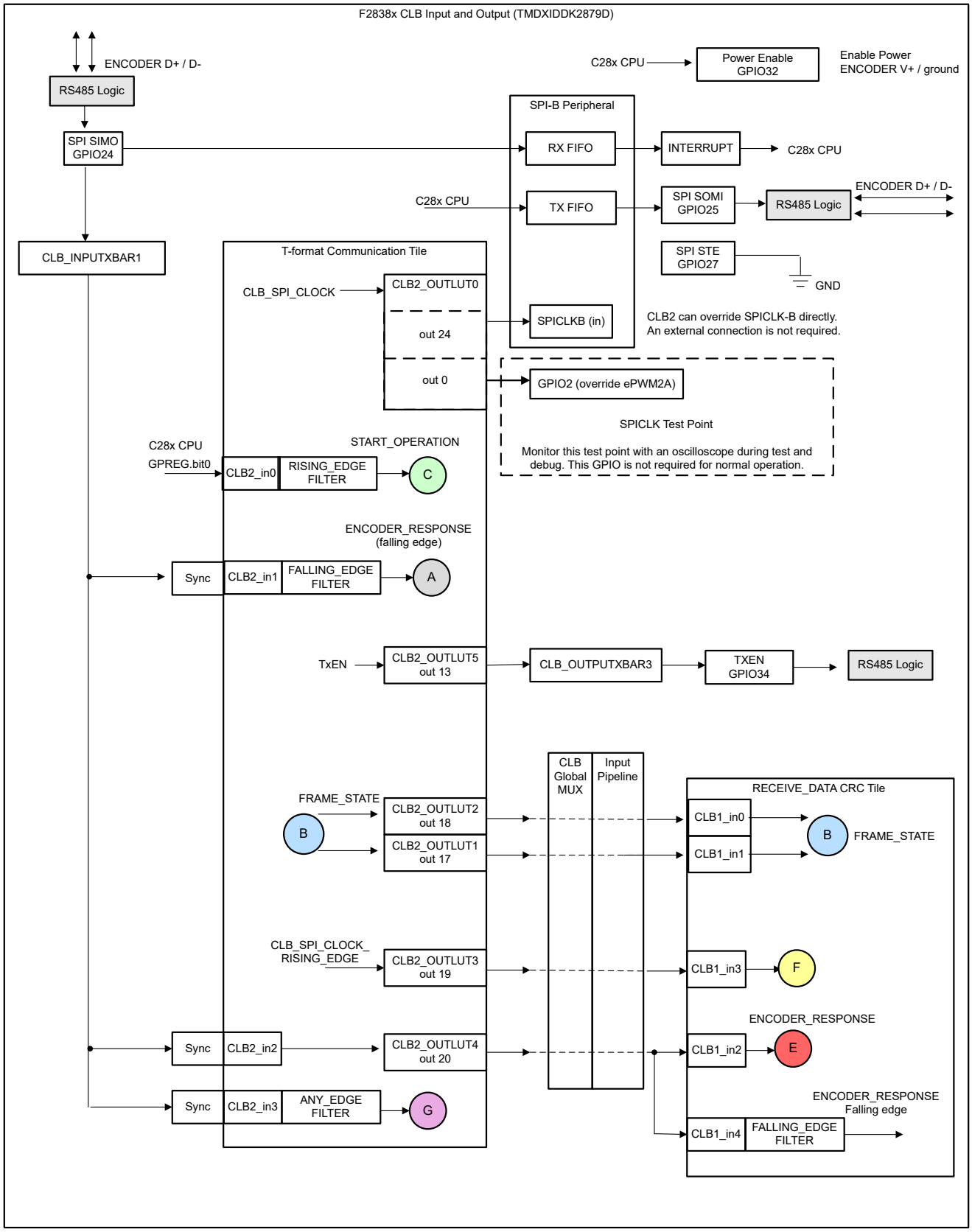


Figure 2-9. F2838x Input and Output for TMDXIDDKF273XD

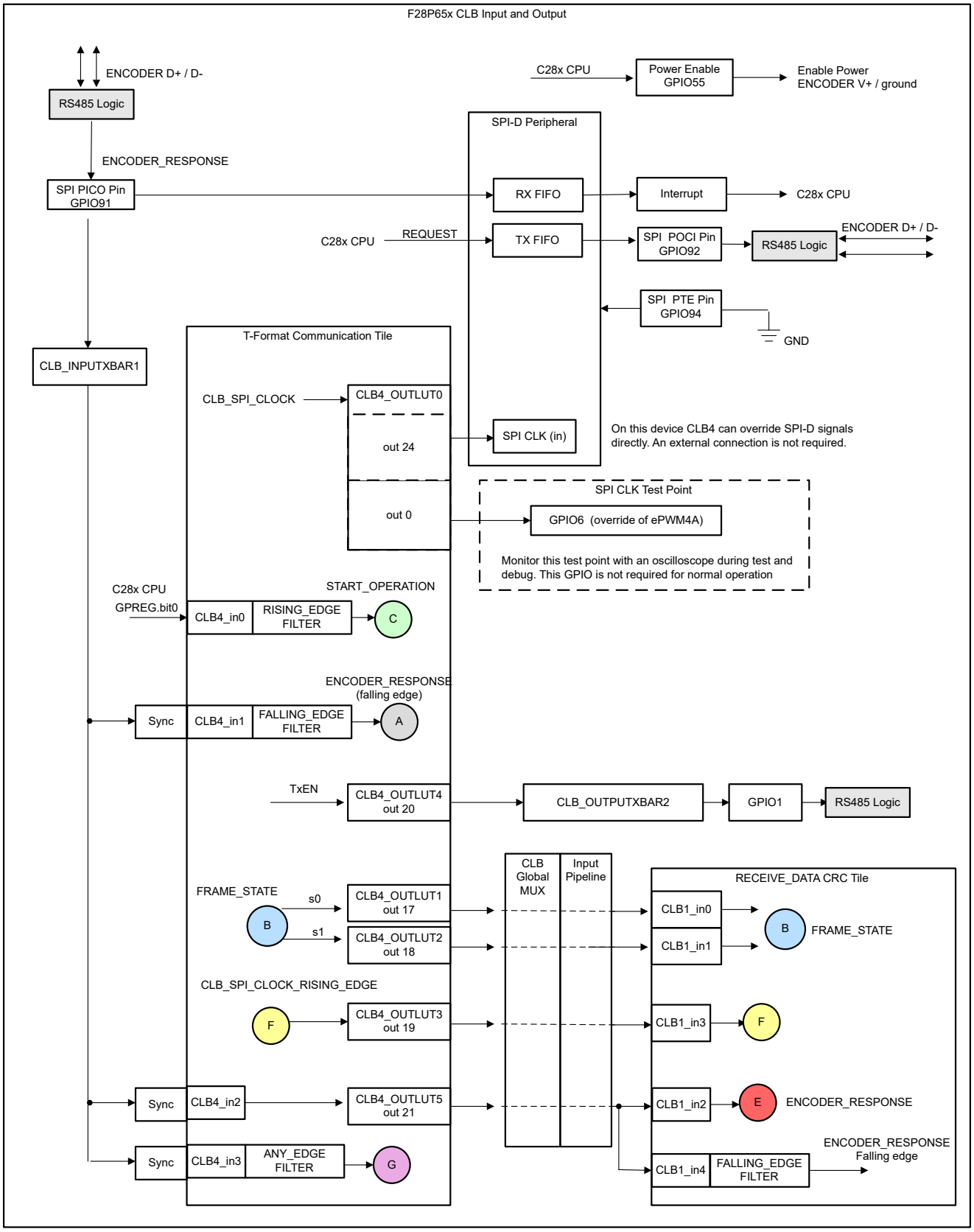


Figure 2-10. F28P65x Input, Output and CLB Usage for BOOSTXL_POSMGR

2.3.6 CLB T-Format Implementation Details

The CLB communications tile is responsible for:

- Clocking the SPI to transmit the request.
- Monitoring the SPI SIMO pin for a response from the encoder.
- Aligning the SPICLK to the incoming response.
- Clocking the SPI to receive the response.

This section describes the design of the communication tile using three different approaches:

1. Visualization of the CLB behavior during each phase of the transaction using waveforms.
2. The CLB tile design including interconnect of the submodules.
3. Using a logic schematic lens.

2.3.6.1 Transaction Waveforms

When implementing a CLB design, first visualizing the required CLB behavior using waveforms can be helpful. To do this, first consider an an example transaction. Recall a T-Format transaction consists of the request transmission plus the encoder's response. A transmission can be broken up into FRAME_STATEs as shown in Figure 2-11. The first step is to map each element of the transaction to a CLB submodule. Table 2-12 shows an example mapping.

Table 2-12. T-Format Transaction to CLB Mapping

Transaction Behavior	CLB Mapping
Track the FRAME_STATE	Finite State Machine (FSM): transitions to a new state given the previous state and current inputs.
Count the clocks generated	A COUNTER: configured to increment a clock edge during the TRANSMIT_DATA and RECEIVE_DATA states. The mode0 input controls when the counter is active and when the counter is halted. Leverage the COUNTER's match values to drive state transitions (TRANSMIT_DATA to WAIT_FOR_START and RECEIVE_DATA to IDLE).
Generate a clock signal of a specific width	This requirement maps to a second COUNTER. Leverage the match values to generate the timing for the the rising/falling edges. A LUT (Lookup Table) then generates the actual edges based on this timing.
Align the clock with the encoder's response	The COUNTER generating the clock can be configured such that the edge transition is properly aligned with the encoder's response.
Only allow the SPI to be clocked during transmit and receive	A LUT blocks the clock when not needed.
Control the TxEN	A LUT leverages the current FRAME_STATE to control the signal.
Tell the CLB to start the transaction	The C28x configures the COUNTER and SPI for the transaction. The CLB GPREG allows the C28x CPU to directly change a CLB input to start a transaction.

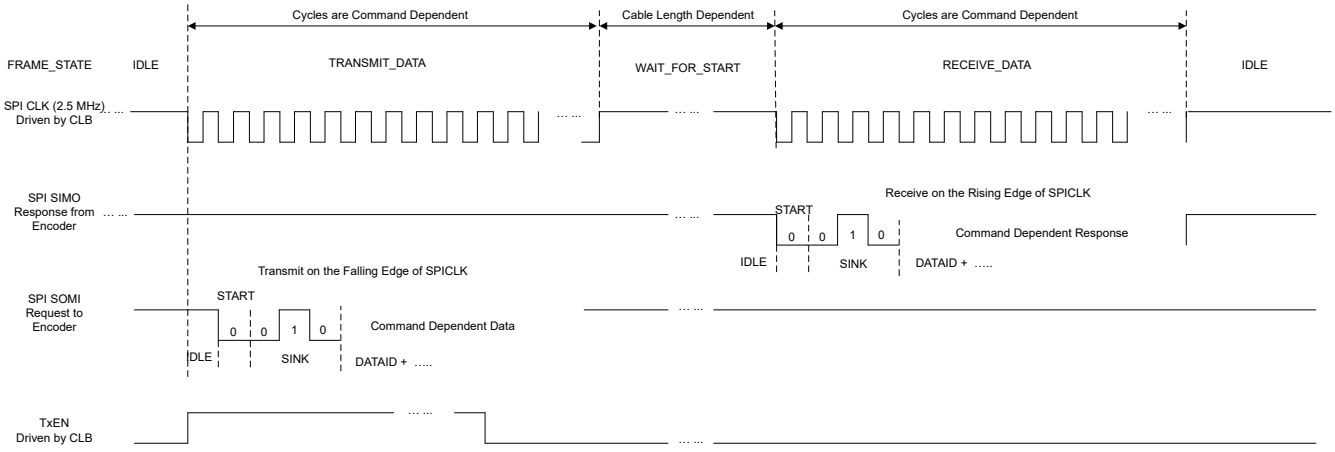


Figure 2-11. T-Format Transaction Example

The next step is visualization of the specific submodule behavior. Start with a quick sketch and then add additional detail as the design develops. Figure 2-12 shows an example waveform.

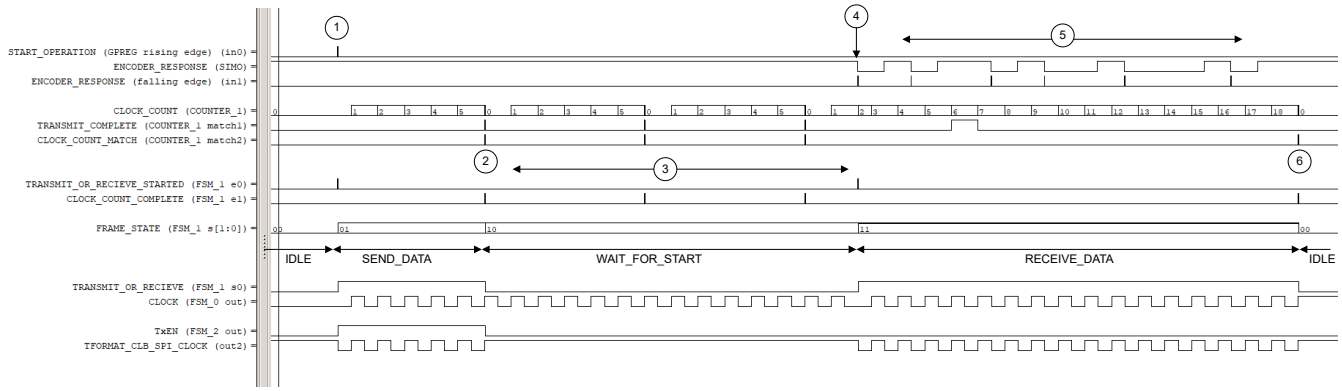


Figure 2-12. CLB Communication Waveform

Note

Figure 2-12 was generated using the CLB SystemC simulation model with a custom input as the encoder's response. The transmitted request is not specifically shown in Figure 2-12. The SPI module sends the request based on the CLB_SPI_CLK during the TRANSMIT_DATA (or SEND_DATA) phase.

Markers 1 - 6 in Figure 2-12 are used in the following sections to describe specific behavior of the design with respect to that marker. The markers are:

1. Transition from IDLE to TRANSMIT_DATA (or SEND_DATA)
2. Transition to WAIT_FOR_START
3. During WAIT_FOR_START
4. Transition to RECEIVE_DATA
5. During RECEIVE_DATA
6. Transition back to IDLE

Following the description of each FRAME_STATE, Section 2.3.6.2 presents the complete tile design. Refer to Section 2.3.6.2 along with Figure 2-12while reading the descriptions of each state.

2.3.6.1.1 IDLE State

During IDLE there is no activity on the interface. The C28x must first setup a request. The T-Format API provides functions to setup each type of request:

- PM_tformat_setupCommandReadEEPROM()
- PM_tformat_setupCommandWriteEEPROM()
- PM_tformat_setupCommandReadoutOrReset()

The _setupCommand function creates a request data packet, loads the data to the SPI TX FIFO, and configures the CLB to generate the number of SPI clocks required to send the request and receive the response. Once setup is complete, the C28x starts the transaction by calling the PM_startOperation() function.

Refer to: Figure 2-2**, marker (1)**.

When the transaction begins, the START_OPERATION signal is pulled high through GPREG. GPREG is the CLB's general purpose register allowing the C28x to directly control the tile's inputs. START_OPERATION remains high for one CLB CLOCK because the CLB's rising edge filter is enabled for the input. At this point, the main state machine (FSM_1) responds by moving the FRAME_STATE from IDLE to the TRANSMIT_DATA state.

2.3.6.1.2 TRANSMIT_DATA State

During the TRANSMIT_DATA (or SEND_DATA) state, the encoder interface sends the request to the encoder. The request is sent from the SPI TX FIFO on the falling edge of the SPICLK. During TRANSMIT_DATA, the CLB:

- Enables transmission through the RS-485 driver by pulling the TxEN signal high.
- Starts generation of the CLOCK signal. CLOCK becomes the CLB_SPI_CLOCK.

- CLB_SPI_CLOCK drives the SPI module clock.
- The CLOCK_COUNT (COUNTER_1) keeps track of the number of CLB_SPI_CLOCKs generated.

The number of CLB_SPI_CLOCKs required to send the transmission was loaded into both the match1 and match2 of the CLOCK_COUNT by the C28x during command setup.

Refer to: Figure 2-2**, marker (2)**.

When the required number of clocks has been reached, the CLOCK_COUNT_COMPLETE (match2) and TRANSMIT_COMPLETE (match1) signals are pulled high. The effect is:

- The main state machine transitions to WAIT_FOR_START
- The CLOCK signal is disconnected from CLB_SPI_CLOCK
- TxEN is driven low to give control of the RS-485 to the sensor or encoder.

2.3.6.1.3 WAIT_FOR_START State

Refer to: Figure 2-2**, marker (3)**.

During the WAIT_FOR_START state, the CLB is monitors the ENCODER_RESPONSE for a falling edge. The falling-edge corresponds to the first start bit in the response from the encoder. The time required can be any number of clock cycles and depends on the state of encoder and the cable length.

During WAIT_FOR_START, the CLOCK_COUNT match and CLOCK_COUNT_COMPLETE signals are ignored and the generated CLOCK is disconnected from the CLB_SPI_CLOCK.

2.3.6.1.4 RECEIVE_DATA State

Refer to: Figure 2-2**, marker (4)**.

On the falling edge of ENCODER_RESPONSE, the FRAME_STATE transitions from WAIT_FOR_START to RECEIVE_DATA.

Refer to: Figure 2-2**, marker (5)**.

During the RECEIVE_DATA, the SPI receives the response from the encoder. The size of the response (number of clocks) depends on the specific request sent during TRANSMIT_DATA. The number of required clocks was configured by the C28x during the command setup. To receive the response:

- The CLB aligns CLB_SPI_CLK to the response. The response is sampled on the rising edge of the clock. Alignment is repeated on each edge of the response by resetting the counter which controls the clock edge placement.
- The CLOCK signal is reconnected to the CLB_SPI_CLK.
- The clock count (COUNTER_1) match2 is adjusted by the HLC. The current count is read, increased by the number of clocks needed to receive the response, and then written back.
- Once the RX FIFO is full, the SPI interrupts the C28x to indicate the response has been received. The C28x calls the specific receiveData function to unpack the data.

Refer to: Figure 2-2**, marker (6)**.

When the number of clocks required to receive the response is reached, the CLOCK_COUNT_COMPLETE signal is pulled high. The CLB returns FRAME_STATE back to IDLE.

This pattern is repeated for each transaction with the encoder.

2.3.6.2 Communication Tile Design

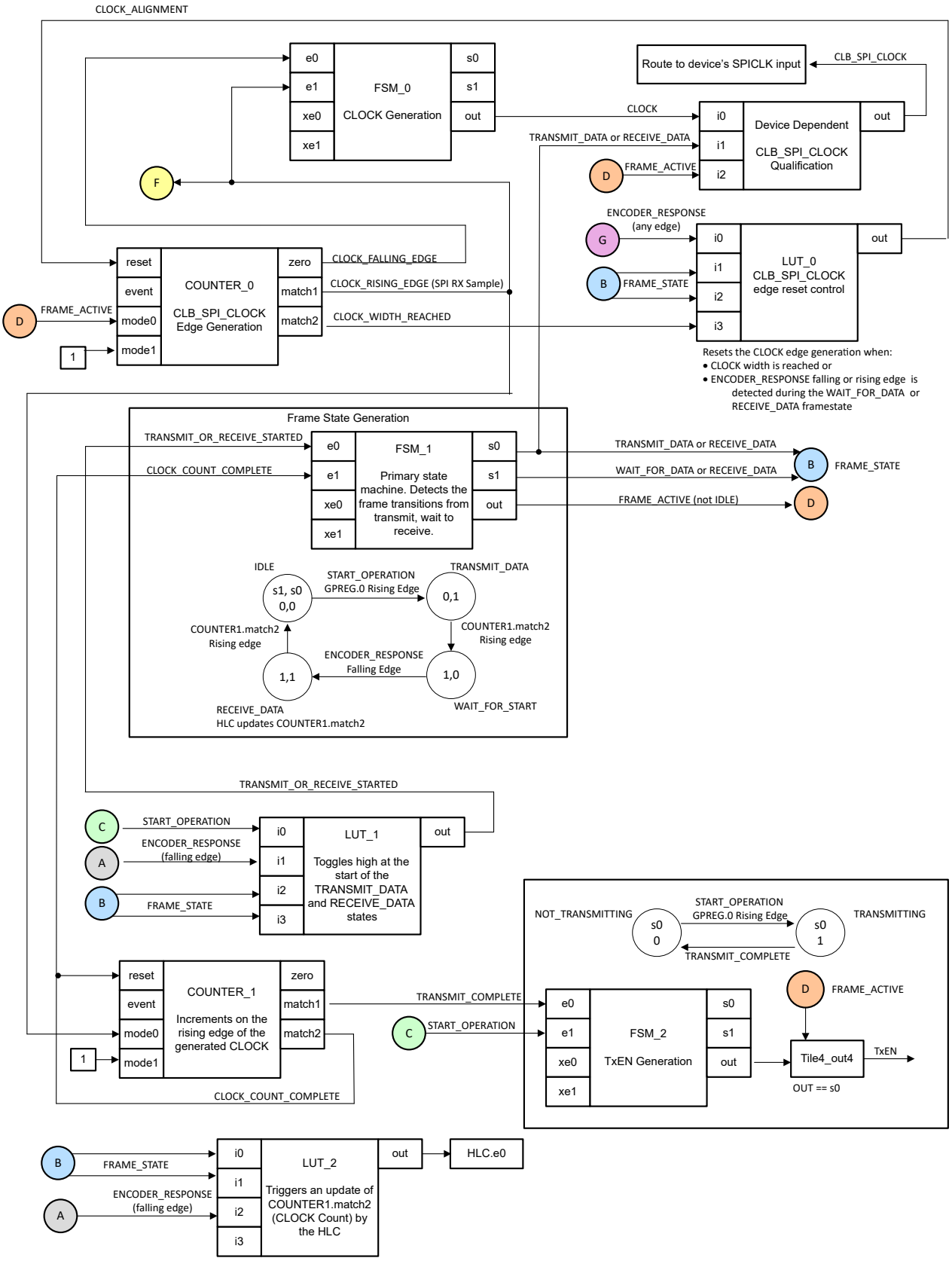


Figure 2-13. T-Format Communication Tile

The communication tile design is shown in Figure 2-13. This section discusses in depth three key components of the design:

1. FRAME_STATE generation (FSM_1)
2. ENCODER_RESPONSE detection (LUT_1)
3. SPI CLK alignment (LUT_0)

Note

In Figure 2-13, a letter in a colored circle indicates an off-page connection described in the input/output diagrams (Section 2.3.5.2).

The equations for other submodules can be found by reviewing the tile's configuration in the CLB tool. Section 2.3.6.3 includes additional information in the form of a schematic for each block.

The FRAME_STATE (FSM_1 s1, s0) transitions are shown in Figure 2-13. To derive the corresponding equations Karnaugh maps are used (Table 2-13 and Table 2-14). The resulting equations are combined by an OR operator and entered into the CLB tool. The equations do not need to be reduced to their simplest form.

Table 2-13. FRAME_STATE FSM_1 Karnaugh Map, State s0

		Current Input (e1, e0) CLOCK_COMPLETE, TX_OR_RX			
		0,0	**0,1**	**1,1**	**1,0**
Previous State s1, s0	**0,0 IDLE**	0 IDLE	1(1) TX	1(1) TX	0 IDLE
	0,1 TX	1(2) TX	1(2) TX	0 WAIT	0 WAIT
	1,1 RX	1(3) RX	1(3) RX	0 IDLE	0 IDLE
	1,0 WAIT	0 WAIT	1(4) RX	1(4) RX	0 WAIT

(1) s0 = (!s1 & !s0 & e0)
(2) s0 = (!s1 & s0 & !e1)
(3) s0 = (s1 & s0 & !e1)
(4) s0 = (s1 & !s0 & e0)

Table 2-14. FRAME_STATE FSM_1 Karnaugh Map, State s1

		Current Input (e1, e0) CLOCK_COMPLETE, TX_OR_RX			
		0,0	**0,1**	**1,1**	**1,0**
Previous State s1, s0	**0,0 IDLE**	0 IDLE	0 TX	0 TX	0 IDLE
	0,1 TX	0 TX	0 TX	1(1) WAIT	1(1) WAIT
	1,1 RX	1(2) RX	1(2) RX	0 IDLE	0 IDLE
	1,0 WAIT	1(3) WAIT	1(3) RX	1(3) RX	1(3) WAIT

(1) s1 = (!s1 & s0 & e1)
(2) s1 = (s1 & s0 & !e1)
(3) s1 = (s1 & !s0)

The OUT signal from FSM_1 is simply an OR of the current states: s1 | s2. This corresponds to any active portion of the frame (not IDLE).

Detection of the encoder's response is another a key component of the design. LUT_1 is responsible for detecting the start of the transaction and the start of the encoder's response. To simplify the design, the following assumptions have been made:

- ENCODER_RESPONSE falling edge never occurs at the same time as START_OPERATION rising edge.

- The START_OPERATION's rising edge only occurs during the IDLE state.

These assumptions are reasonable given the encoder only responds to a transmission initiated from the C28x and the C28x controls START_OPERATION. These assumptions result in the equation: out = (i3 & i2) & i1 | i0:

- If FRAME_STATE == WAIT_FOR_START (i3 & i2) and ENCODER_RESPONSE falling edge (i1) then output goes high.
- If START_OPERATION rising edge (i0), then the output goes high
- Otherwise the output is low.

The encoder's response can come at any time due to cable delays. To to read the response correctly, the CLB_SPI_CLK must be correctly aligned. LUT_0 is responsible for both the clock alignment and the width of the clock. Both alignment and clock width are achieved by LUT_0 resetting COUNTER_0 at the appropriate time. COUNTER_0 match values control the CLB_SPI_CLOCK edge timing.

- Clock width: reset COUNTER_0 when CLOCK_WIDTH_REACHED (i3) is high.
- Align the clock: If the FRAME_STATE is WAIT_FOR_START (i2 & !i1) or RECEIVE_DATA (i2 & i1), then reset COUNTER_0 on ENCODER_RESPONSE any edge (i0).

This results in the equation: LUT_0 out = i3 | (((i2 & !i1) | (i2 & i1)) & i0).

The equations for other submodules can be examined by reviewing the tile's configuration in the CLB tool. Section 2.3.6.3 includes additional information in the form of a schematic for each block.

2.3.6.3 Logic View

The following figures show the same information as Section 2.3.6.2 but through a logic schematic lens. Specifically:

- Figure 2-14 and Figure 2-15 show the contents of the CLB blocks using logic gates.
- Figure 2-16 uses this logic to show how the main state machine controls other blocks.
- Figure 2-17 traces a couple of simple CLB output signals starting from their inputs and passing through some associated logic.
- Figure 2-18 traces the Clock to SPI output starting from Input1 and passing through LUT_0, FSM_0, Counter_0 all the way to the Ouput_LUT_0 as controlled by 3 outputs from FSM_1.
- Figure 2-19 traces the Transmit Enable output starting from Input0 and Input1, and passing through LUT_0, Counter_0, Counter1 and FSM_2 as controlled by 3 outputs from FSM_1.

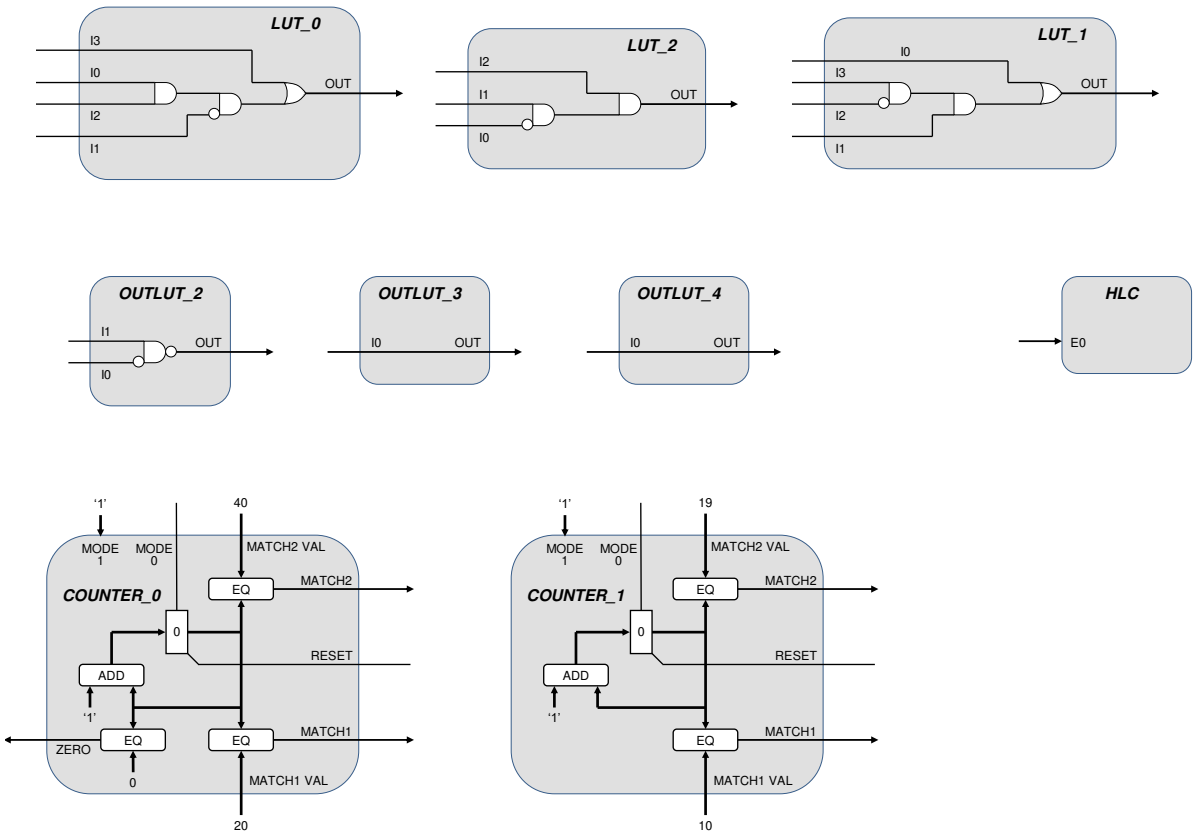


Figure 2-14. LUTs, OUTLUTs, and Counters

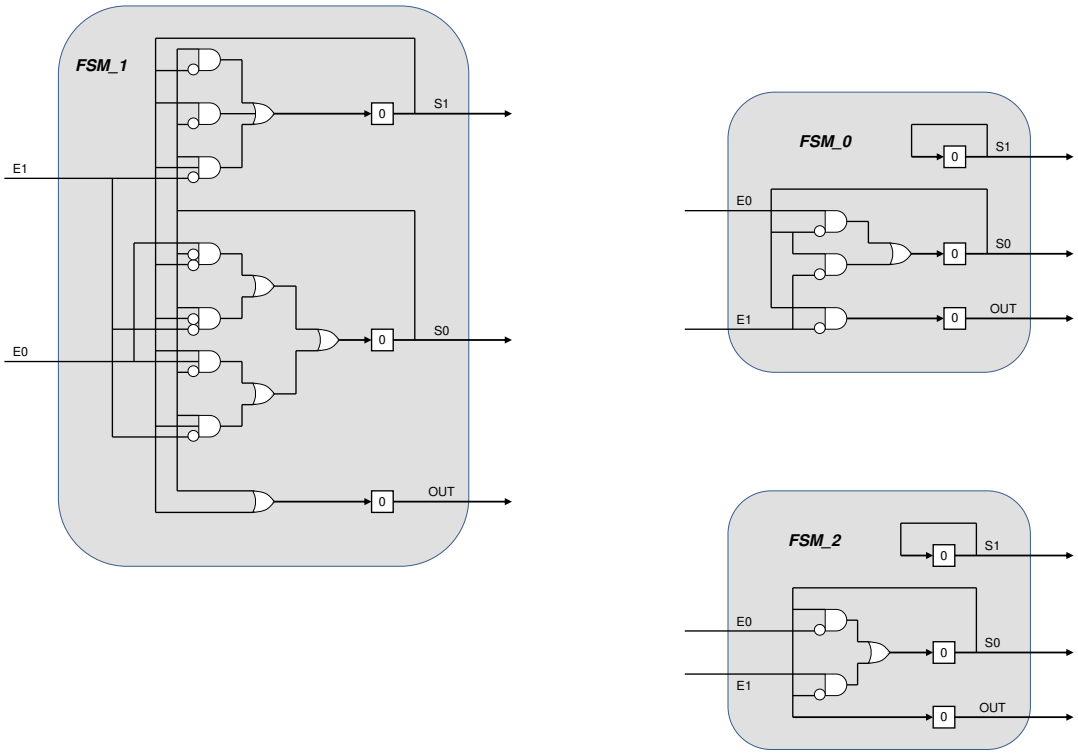


Figure 2-15. Finite State Machines

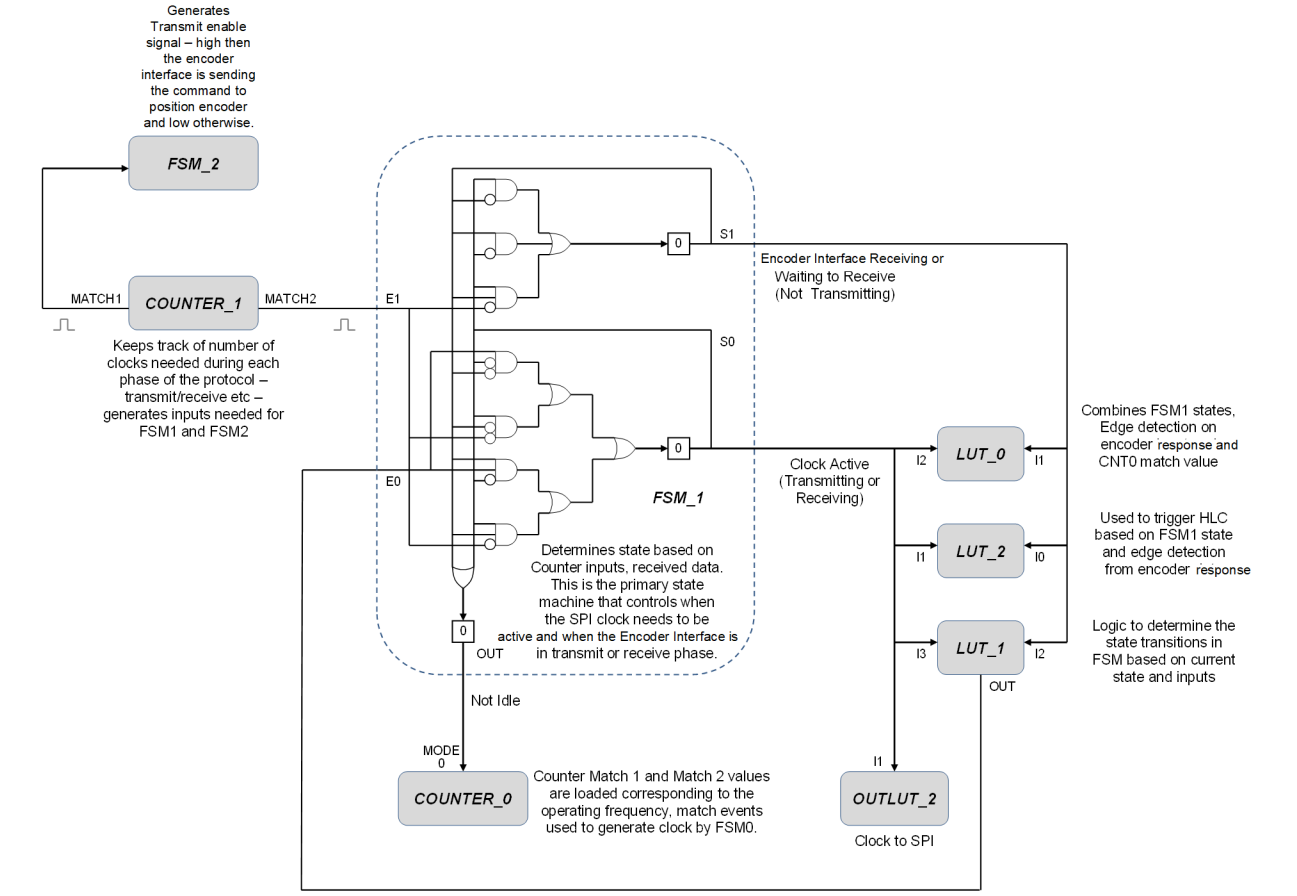


Figure 2-16. The Main State Machine

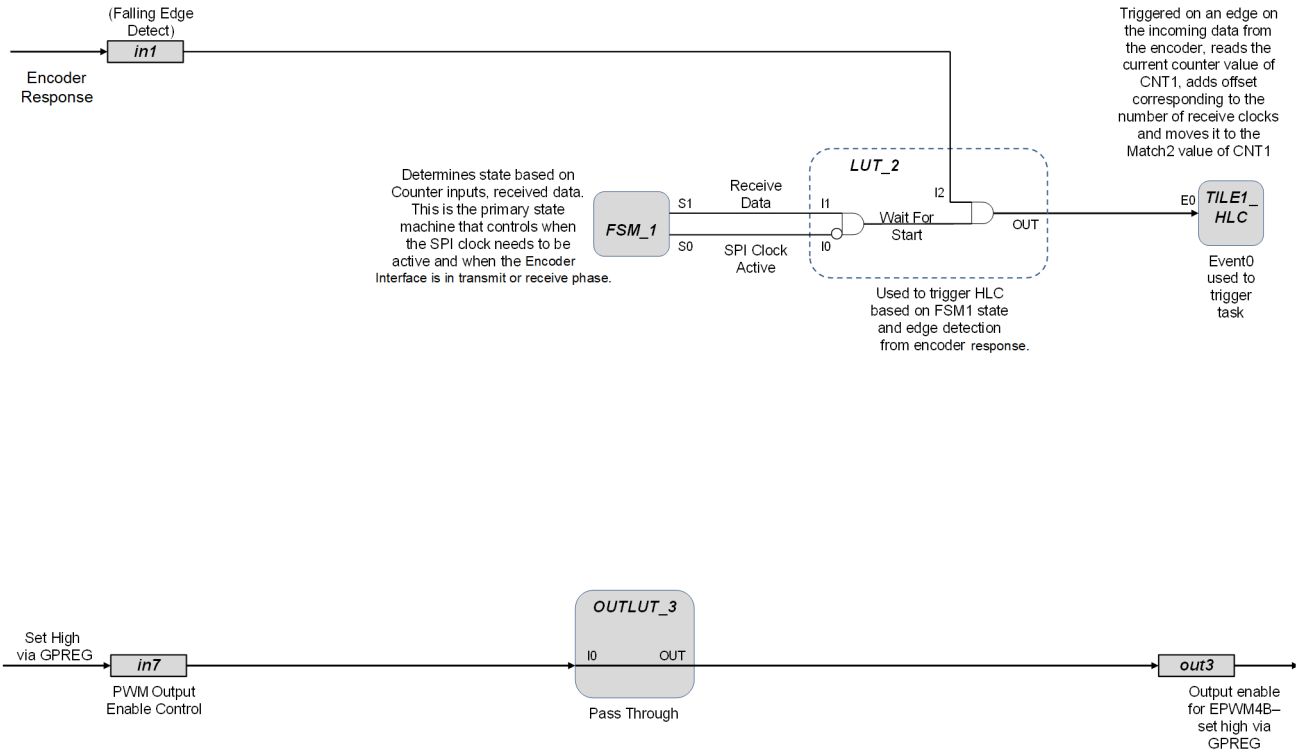


Figure 2-17. CLB Outputs – HLC Event0 and EPWM Output Enable

Note

Only the F2837xD design overrides the EPWM outputs.

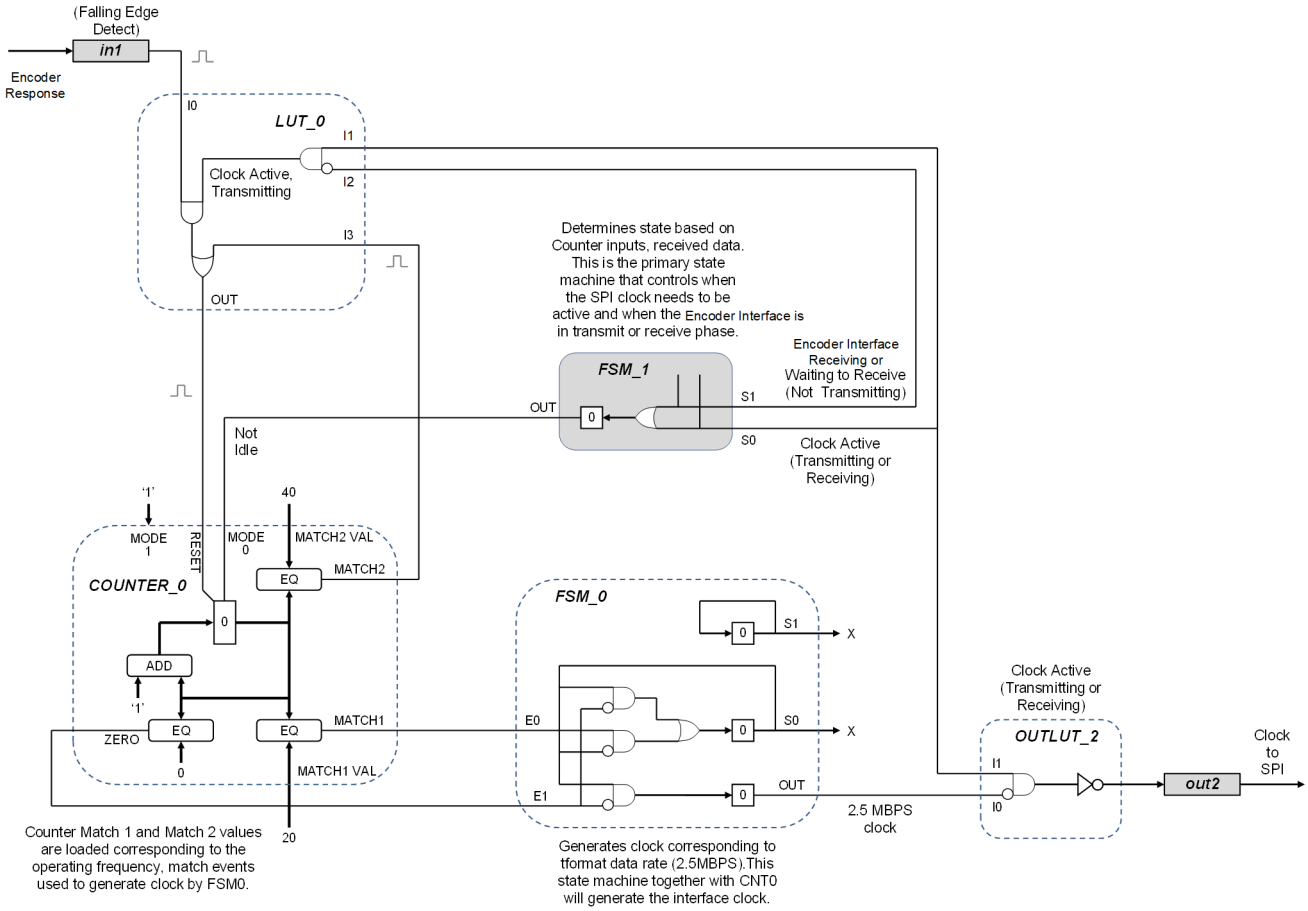


Figure 2-18. CLB Outputs – Clock to SPI

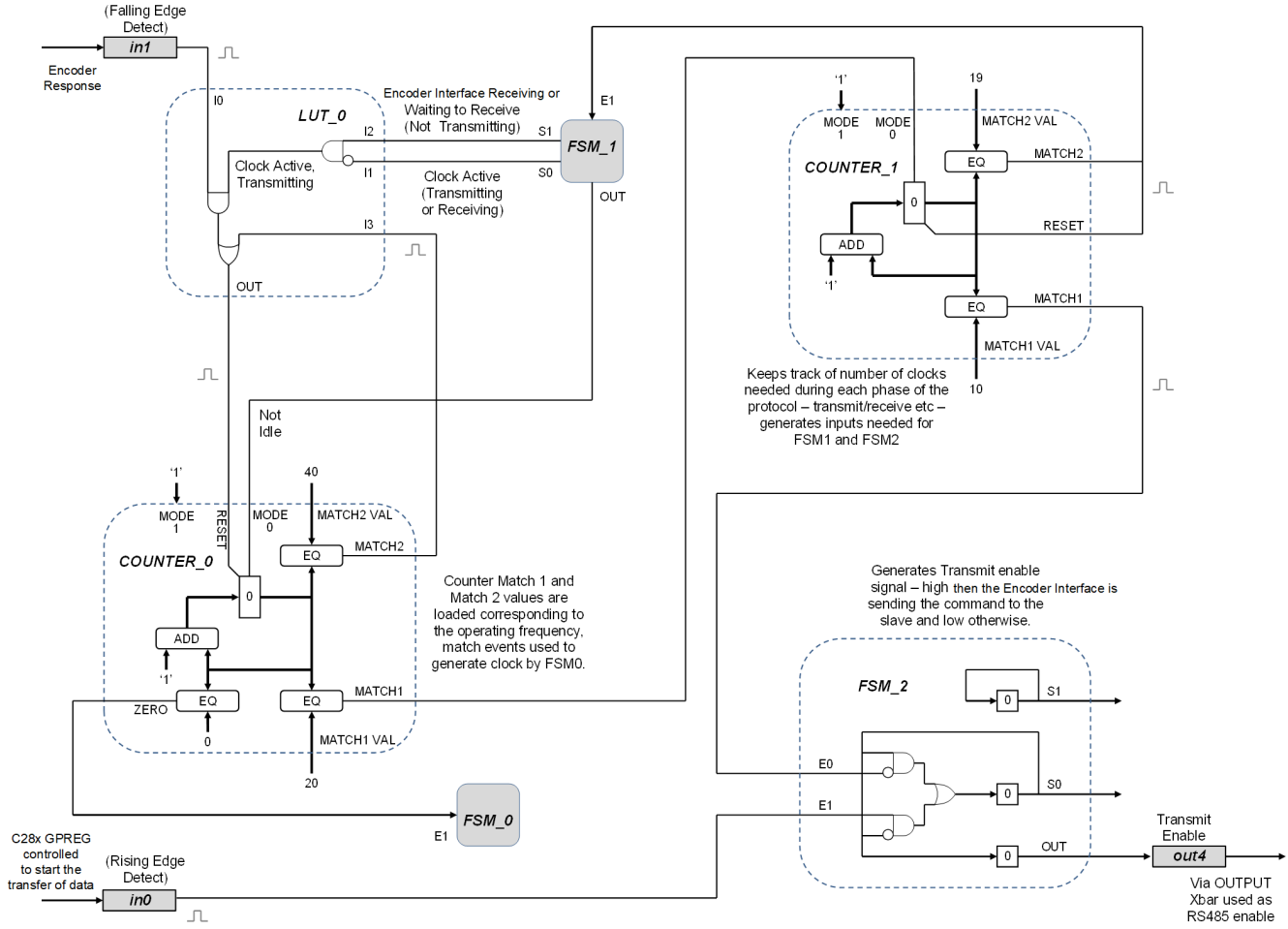


Figure 2-19. CLB Outputs – RS485 Enable

2.3.7 CLB Receive Data CRC Implementation

As explained in Section 2.3.5.1, CLB type 2 or later has the capability of calculating the response-data CRC as the response is being received, or on-the-fly. In this case, required signals are connected to a second tile to perform the CRC calculation. The connections between tiles are documented in Section 2.3.5.2.

To generate a CRC, a counter is configured as a Linear Feedback Shift Register (LFSR). Data received is fed to the event input of the LFSR. When data is valid, a shift is applied through the LFSR's event input. This design requires knowing when the data is valid so the shift can be properly applied. Recalling the protocol overview in Section 2.3.1, the criteria for valid data is:

- FRAME_STATE is RECEIVE_DATA
- The current bit is one of the 8-bits of data within the field. That is, the start and delimiter are skipped. This is visualized in Figure 2-21.
- The field is not the CRCField. The CRC is always the last field in the encoder's response.

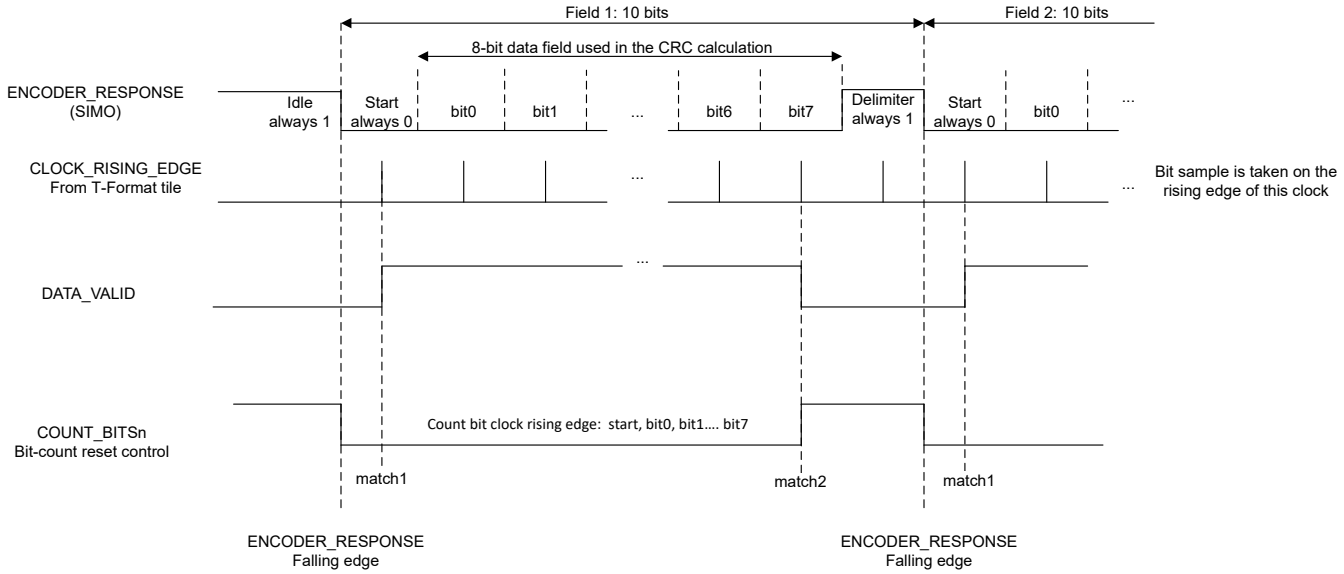


Figure 2-20. CRC Field Data Valid

The specific requirements map to CLB functionality as shown in Table 2-15.

Table 2-15. CRC Generation to CLB Mapping

CRC Functionality	CLB Mapping
Include only 8-bits of data (skip start and delimiter)	COUNTER module to count the bits within a T-Format field. The match values indicate the first valid bit, and the last valid bit as shown in Figure 2-20. An FSM then determines if the shift is applied to the LFSR based on the counter match outputs.
Include only valid fields (skip the CRC).	COUNTER module that increments at the start of each field during the RECEIVE_DATA state. When the CRCField is reached, the match output is asserted.
Control the shift (mode0) of the LFSR	Use a LUT to determine if the data is valid based on the bit count, and field count. If valid, the LUT applies a shift pulse to the LFSR.

Figure 2-21 shows an example waveform for Data ID3 CRC generation. Figure 2-22 shows the CRC tile implementation. The equations for the submodules can be viewed using the CLB Tool.

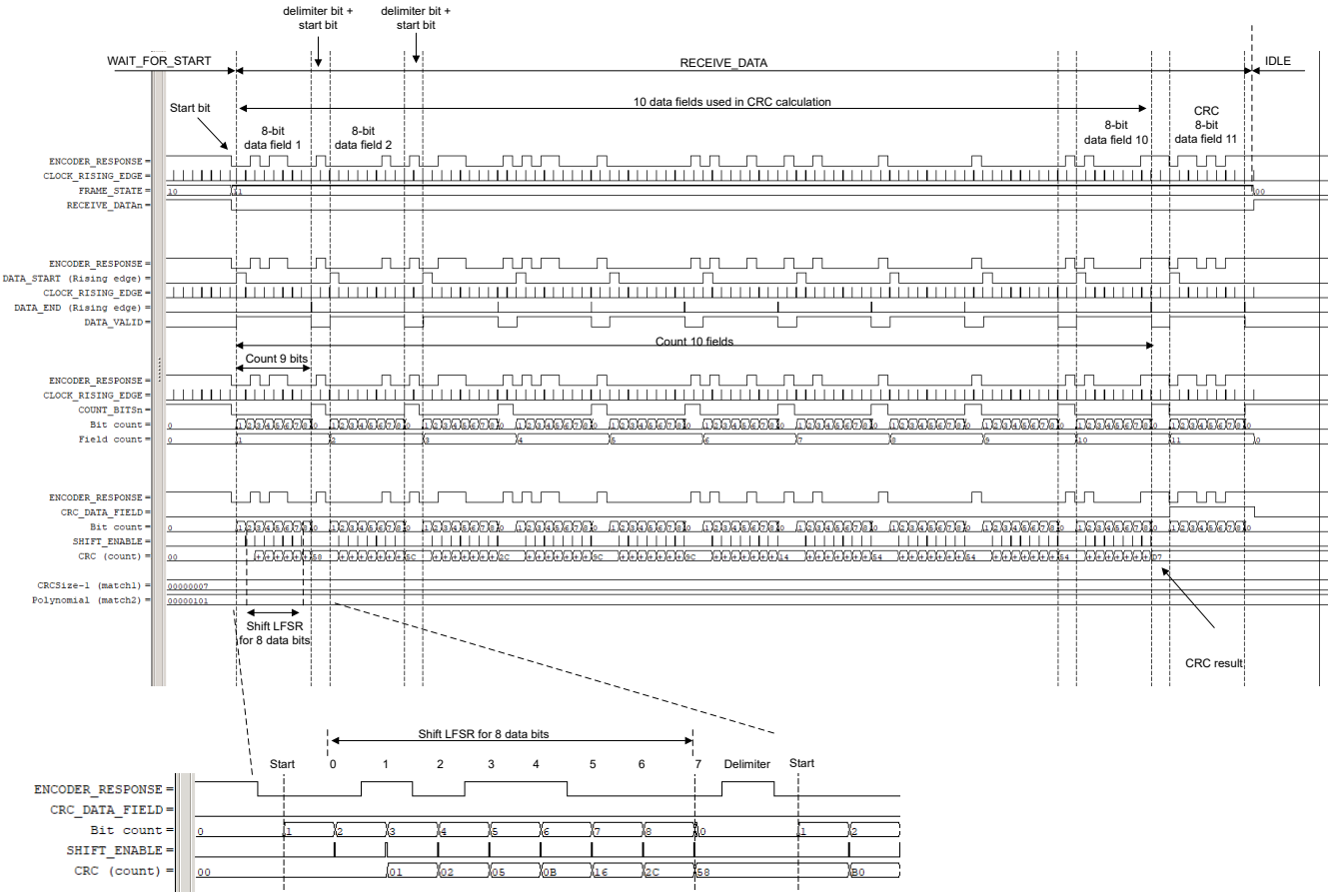


Figure 2-21. CRC Waveform Data ID3

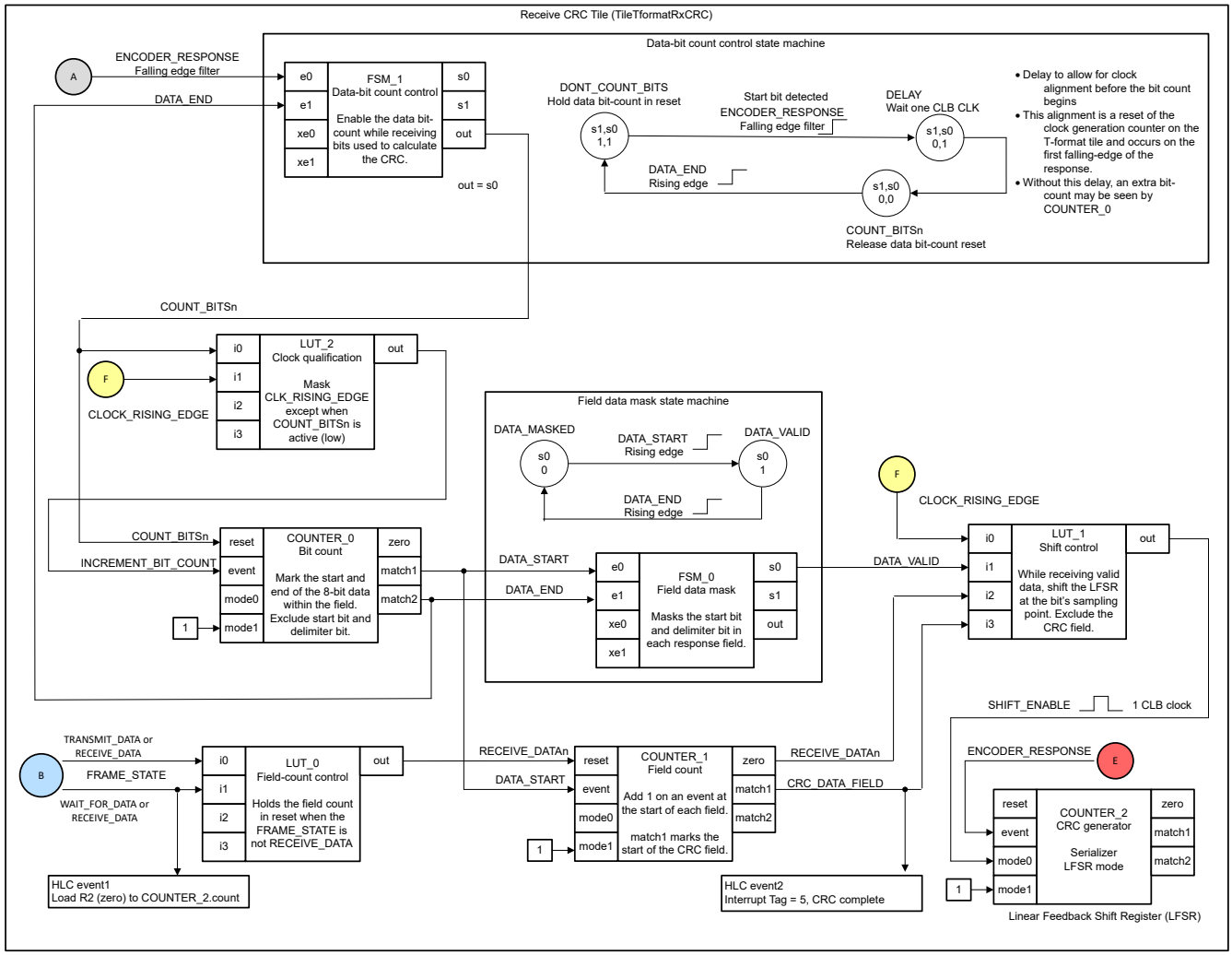


Figure 2-22. CLB Receive Data CRC Generation Tile

2.3.8 PM T-Format Encoder Interface Library

The PM T-Format encoder interface library provides:

- The CLB logic implementation described in Section 2.3.6 and Section 2.3.7.
- A well-defined application programming interface (API) to enable C2000 devices to communicate with T-Format position encoders

This section provides a high-level overview of the functions provided by the API.

2.3.8.1 PM T-Format Reference Implementation Commands

The commands described in Section 2.3.1 are supported by the T-Format reference implementation. The example provides some error checking. Additional features are left to the system developer. Details of the T-Format protocol can be obtained from Tamagawa.

2.3.8.2 Functions Supported in PM T-Format Reference Implementation

The PM T-Format application programming interface (API) enables the C28x to communicate with T-Format encoders. For a detailed description of the API, refer to the: ***C2000 T-Format Encoder Interface Software Guide*** (html, pdf).

The software guide documents:

- Communication demonstration project
- T-Format application programmer interface (API)
- Incorporating the library into your own solution
- Migration from previous versions

Table 2-16 provides a high-level overview of the API.

Table 2-16. Functions in T-Format Reference Implementation

API Function Type	DESCRIPTION
Run-time Functions	
Command Setup	Set up the SPI and CLB for a given request to be transmitted and response received. Support is provided for all T-Format commands: readout, reset and EEPROM commands
Start Operation	This function starts the CLB state machine transition to TRANSMIT_DATA. This initiates the transfer of the request.
Receive Data	Request-specific functions for unpacking and populating the T-Format data structure with the data received from the encoder.
Update Encoder Parameters	Functions to update the position, turns or encoder ID based on the data received from the encoder.
Initialization Functions	
Setup Peripheral	Setup for SPI, CLB, and other interconnect XBARs for T-Format are performed with this function during system initialization. This function must be called after every system reset. No T-Format transactions is performed until the setup peripheral function is called.
Generate CRC Table	Only required if the C28x calculates the CRC using a look-up table. Generates a table of 256 entries for a given CRC polynomial (polynomial) with a specified number of bits (nBits).
Set Frequency	Scales the CLB_SPI_CLK to match the T-Format frequency. The scaling is dependent on the CLB clock.

3 Hardware, Software, Testing Requirements, and Test Results

3.1 Hardware

To experiment with the TIDM-1011, the following hardware components are required:

- TIDM-1011 BoosterPack (also known as the BOOSTXL-POSMGR)
- External, 5-V, DC power supply (see Table 1-1)
- A Supported LaunchPad featuring a C28x device with the Configurable Logic Block (CLB). Refer to Table 2-1 for a list.
- USB-B to A cable
- T-Format Absolute Encoder from Tamagawa (for example, TS5700N8501)
- 4-pin cable from Tamagawa – length as required by the application (maximum 100 m)
- Custom adapter to connect Tamagawa, 4-position, female-terminated cable to wire leads adapter
- PC with Code Composer Studio (CCS) v12.0.0, or greater, installed

Note

Not all Tamagawa absolute encoders use T-Format for communication. To use the provided software as-is, the format must be T-Format.

3.1.1 TIDM-1011 Jumper Configuration

The Position Manager BoosterPack is expected to be plugged onto the LaunchPad site 2, as shown in Figure 3-4.

Figure 3-1 shows the jumper configuration for TIDM-1011 board.

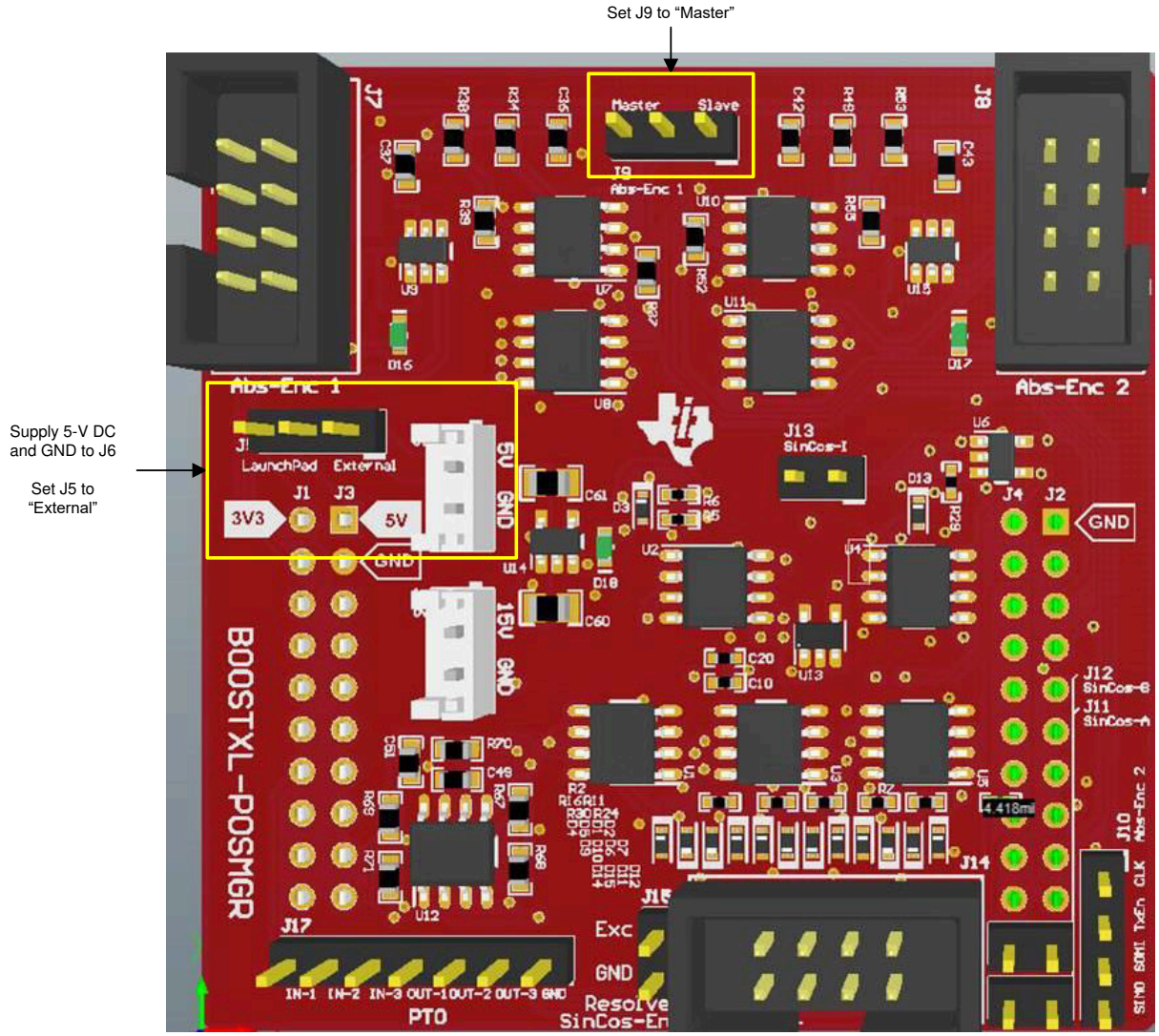


Figure 3-1. TIDM-1011/Position Manager BoosterPack Jumper Configuration

Table 3-1 lists the jumper configuration for the TIDM-1011 board.

Table 3-1. TIDM-1011 Board Jumper Details

JUMPER	FUNCTION	POSITION
J5	TIDM-1011, 5-V, power-plane source selection	*External* (1)
J9	Abs-Enc-1, master-slave mode selection	*Master* (2)
J11	Sine-Cosine, encoder-A signal enable	Open
J12	Sine-Cosine, encoder-B signal enable	Open
J13	Sine-Cosine, encoder-index signal enable	Open

(1) This configuration requires providing an external power source to J6, as shown in Figure 3-1.
(2) Do not use the slave mode option. There is an error in the bootsterPack logic for this mode.

3.2 Software

This section provides an overview of the software used by TIDM-1011. For comprehensive documentation, refer to the *C2000 T-Format Encoder Interface Software Guide* (HTML, PDF).

The software guide includes:

- Documentation of the system demonstration code
- Documentation of the T-Format application programmer interface (API)
- Incorporating the library into your own solution
- Guidance for porting the solution to C28x CPU2
- Change history
- Migration from previous versions of the library

Figure 3-2 shows the software architecture implemented in this reference design. The software is implemented in a modular and portable manner. The main components include the C2000 Driver Library, the T-Format Encoder Interface Library, the SysConfig GUI Device Configuration Tool, and the CLB Tool.

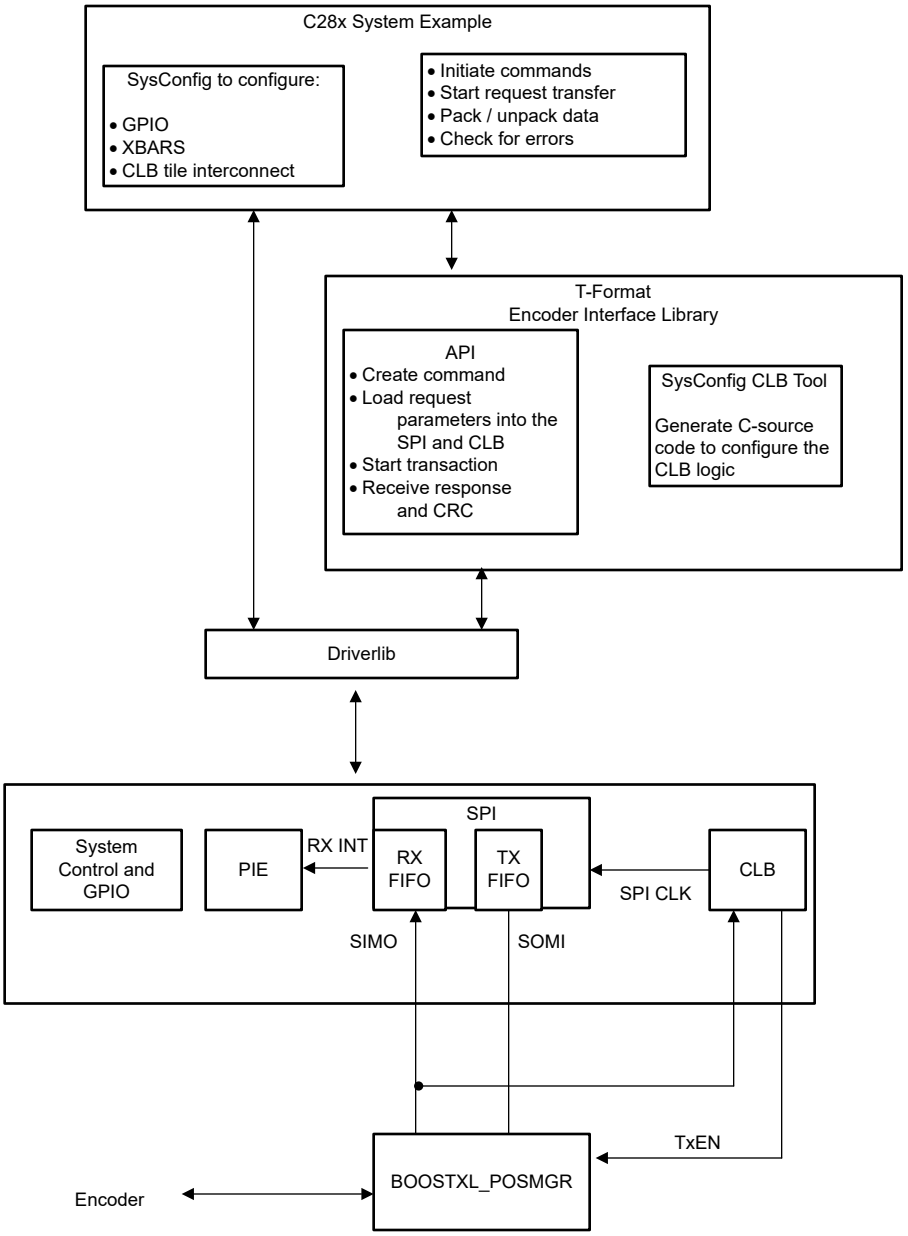


Figure 3-2. T-Format Reference Software Architecture

A flowchart for the C2000 T-Format communications demo is shown in Figure 3-3. The example application configures the C2000 device, creates command request data packets, initiates commands, unpacks responses, and checks the CRC.

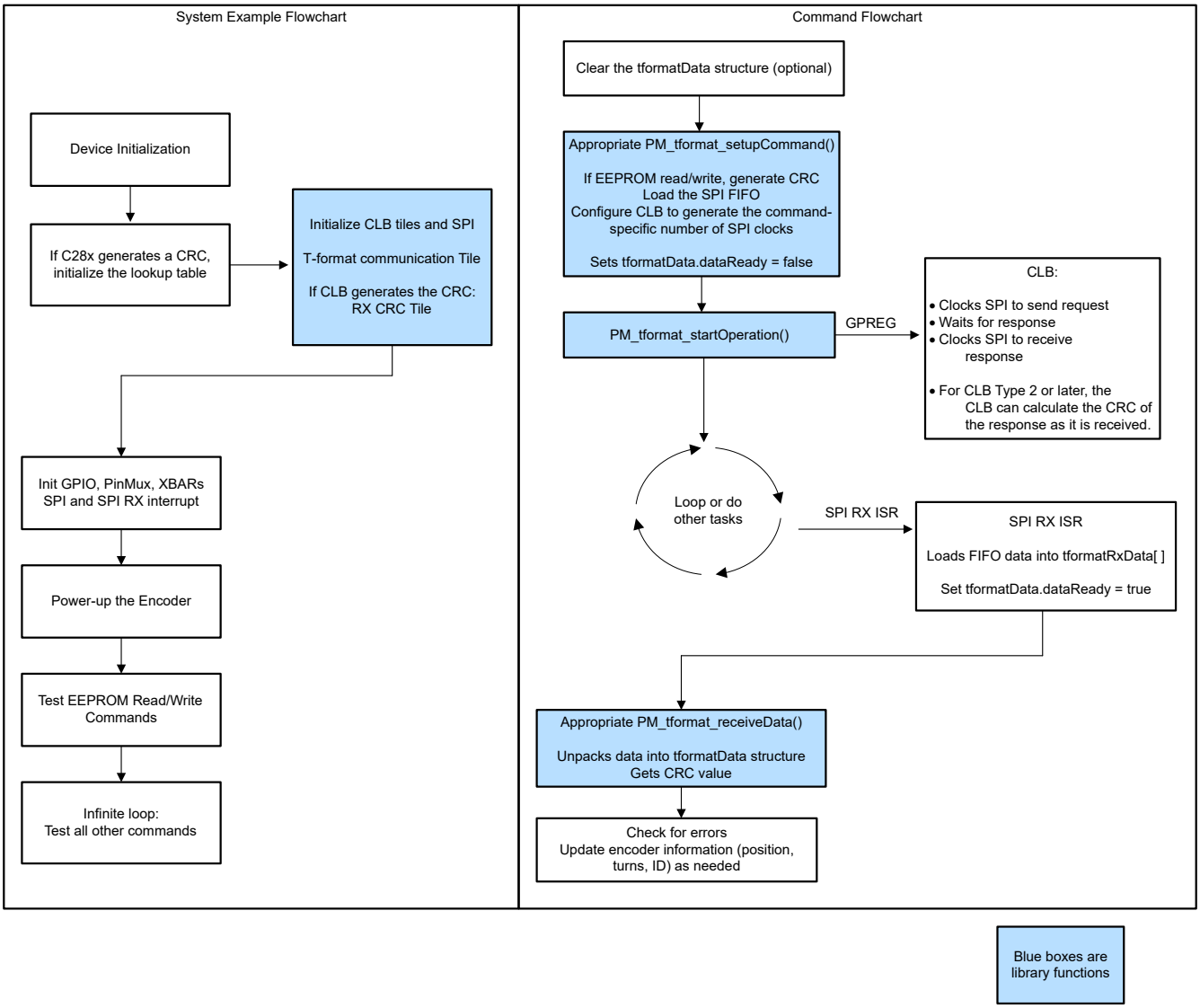


Figure 3-3. T-Format Reference Software Flow Diagram

3.2.1 C2000 Driver Library (DriverLib)

The C2000 Driver Library (Driverlib) is a set of low-level APIs for C2000 device families. Driverlib provides easy-to-use function calls for configuring memory-mapped peripheral registers. Full source for Driverlib is provided within C2000Ware and the C2000Ware Motor Control SDK. For more information, refer to the DriverLib section of the *C2000 Software Guide*.

3.2.2 C2000 SysConfig

C2000 SysConfig is a graphical user interface tool for configuring the C2000 Real-Time Control MCUs. SysConfig auto-generates embedded software that interfaces to Driverlib. In this reference design, the SysConfig tool is used to generate code to configure the SPI, GPIO, INPUTXBAR/OUTPUTXBAR and CLB MUX. For more information, refer to the *C2000 Academy: SysConfig module*.

3.2.3 C2000 Configurable Logic Block Tool

The C2000 CLB Tool enables configuration of the CLB Logic through a graphical interface. The CLB Tool is an easy-to use GUI built into Code Composer studio and makes use of the C2000 SysConfig plug-in. In this reference design the CLB Tool is used to configure the tile for the T-Format Encoder Interface as described in the design description. For more information, refer to the *C2000 Academy: Configurable Logic Block module*.

3.2.4 Installing Code Composer Studio™ and C2000WARE-MOTORCONTROL-SDK

1. Install CCS v12.0.0 or later, if it is not already on the PC.
2. Install C2000WARE-MOTORCONTROL-SDK v4.01.00.00 or later, if it is not already installed on the PC
3. After installation, refer to the *C2000 T-Format Encoder Interface Software Guide* (html, pdf) for further instructions.

Note

To build the examples, only the above software is required. To re-build the CLB-based libraries, the CLB Tool is also required. This tool is included in Code Composer Studio (sysconfig) and the C2000Ware sub-component of the SDK (support utilities). To run CLB-based simulations requires installation of additional tools which are documented in the *CLB Tool User's Guide*.

3.2.5 Locating the Reference Software

The software included in this reference design consists of two parts:

- A system example that illustrates the usage of the T-Format encoder interface. The location of the example project source files is shown in Table 3-2.
- The T-Format encoder interface library. The location of the library source files is shown in Table 3-3.

For comprehensive documentation, refer to the "C2000 T-Format Encoder Interface Software Guide" (html, pdf).

Table 3-2. Location of System Examples

`C:\ti\c2000\C2000Ware_MotorControl_SDK_[version]\`	Default install location for the SDK. (`[SDK]`)
`[SDK]\solutions\boostxl_posmgr\`	Device-specific base install directory (`[pm_base]`)
`[pm_base]\[device]\ccs\tformat`	Code Composer Studio (CCS) projectspec files. Used to import the project into your CCS workspace.

Table 3-3. Location of the T-Format Encoder Interface Library

`C:\ti\c2000\C2000Ware_MotorControl_SDK_[version]`	Default install location for the SDK. (`[SDK]`)
`[SDK]\libraries\position_sensing\tformat`	Library base install directory (`[lib_base]`)
`[lib_base]\ccs\[device]`	Code Composer projectspec file for the reference library. Use these projects to re-build the library for each device.

3.3 Testing and Results

This section details the test procedure, results, and benchmarks. A troubleshooting guide is also provided.

3.3.1 Hardware Configuration

1. Ensure that the jumper configuration of the TIDM-1011 board is as described in Table 3-1.
2. Connect the TIDM-1011 board to the LaunchPad using the BoosterPack connector (J5 to J7 and J8 to J6). Ensure the TIDM-1011 device is connected to site two of the LaunchPad, as shown in Figure 3-4.

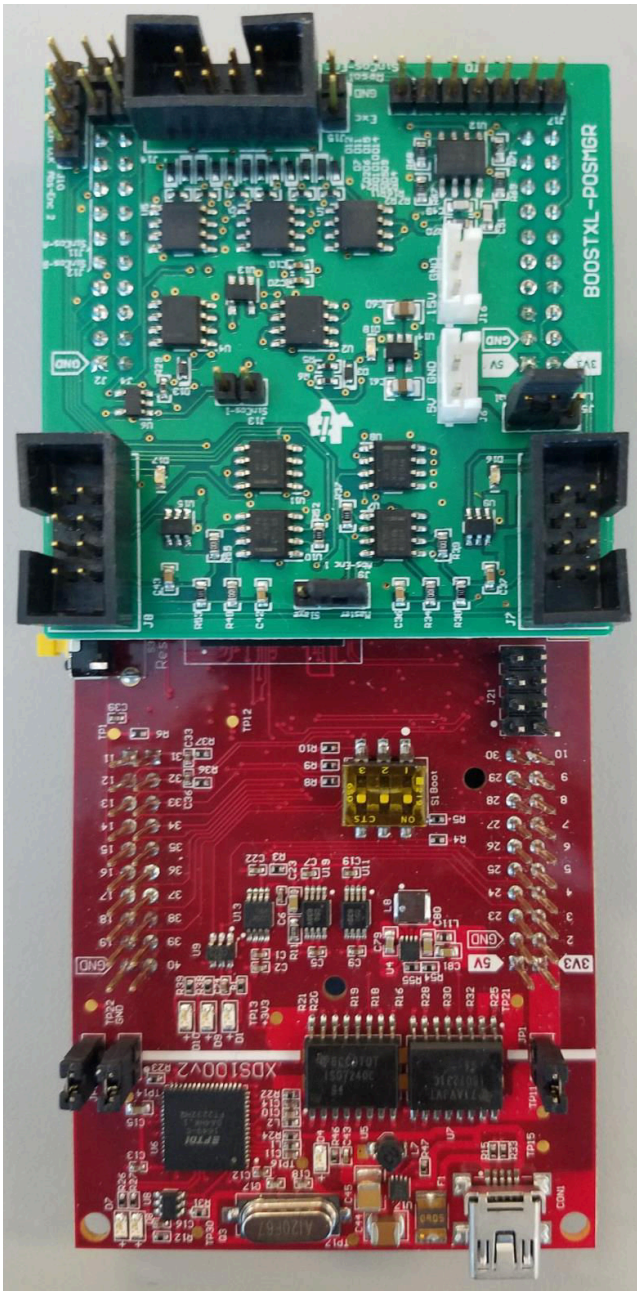


Figure 3-4. Position Manager BoosterPack Connected to Site Two of LaunchPad™

3. Connect the USB cable to the LaunchPad.
4. Set up the connection to the encoder.
   a. Prepare an adapter to connect the Tamagawa cable to the T-Format interface using a 8-position female to wire the leads adapter (see the BOM for the header used for the encoder connector, J7).
   b. Insert the header of the adapter, created in the previous step, to connect to Abs-Enc-1 (J7). The female end of the Tamagawa cable connects to the encoder. Figure 3-5 shows the pinout of J7.

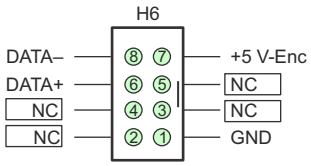


Figure 3-5. Abs-Enc-1 (J7) Pinout on TIDM-1011 Board

5. Supply 5-V DC and GND to J6, as shown in Figure 3-1. The board should now look like Figure 3-6. LED D18 should be lit, which shows that the board has power.

Note

For some encoders the BoosterPack may not provide an adequate current at power up. If the encoder fails to respond, try connecting a power supply external to the BoosterPack to the encoder. If this is done, connect a common ground to the BoosterPack.

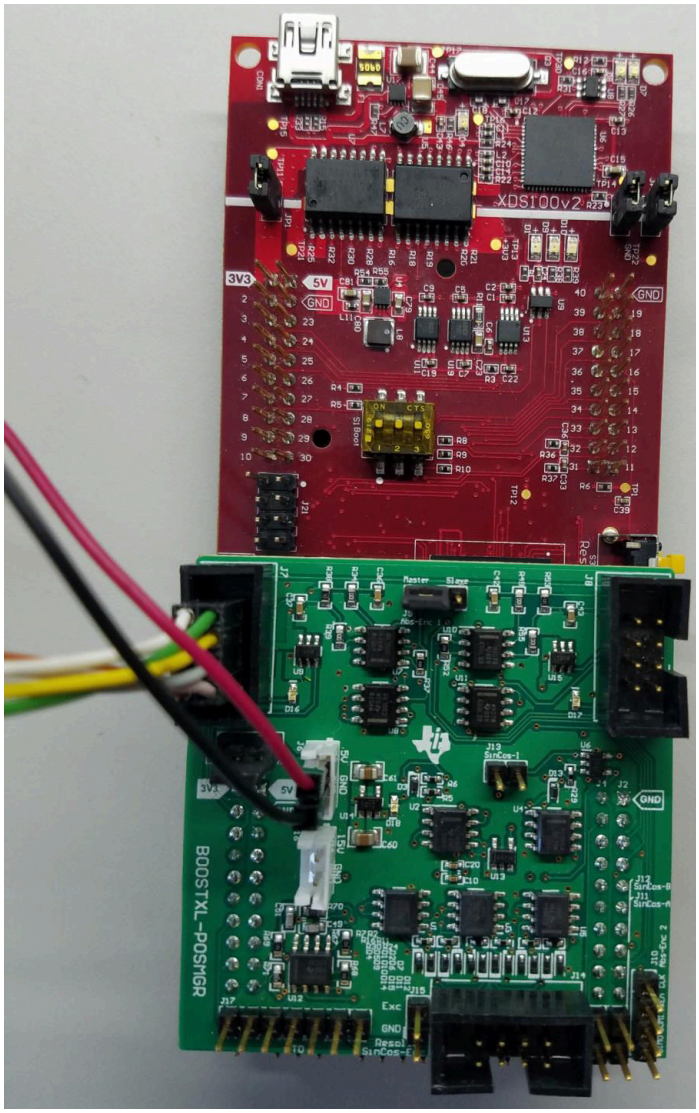


Figure 3-6. Position Manager BoosterPack Powered On and Connected to Tamagawa Encoder

3.3.2 Building and Loading Project

Follow the instructions in the ***C2000 T-Format Encoder Interface Software Guide*** (html, pdf) to load and run the system solution. Refer to the T-Format System Solution section of the software guide.

These directions include:

- Importing the projects into Code Composer Studio (CCS) for your device
- Configuring the library and system example
- Selecting the build configuration
- Populating the watch window
- Running the code

3.3.3 Running Code

The T-Format System Solution is a communications-only demonstration. The demo sends a command, receives the response, and checks for errors. This pattern is repeated for each of the T-Format commands. While running the demo, you can monitor output signals of the MCU with a logic analyzer or scope. Example transactions for Data ID 2 and Data ID D are shown in Figure 3-7 and Figure 3-8.

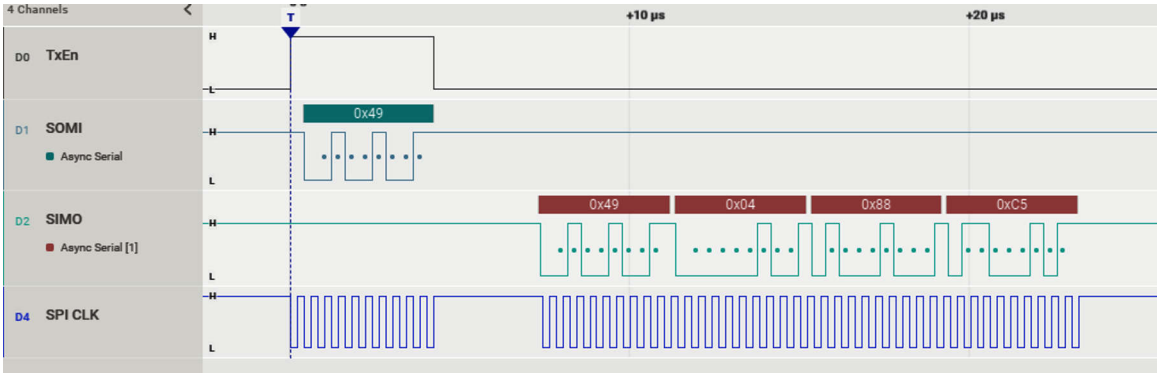


Figure 3-7. Data ID 2 Waveform

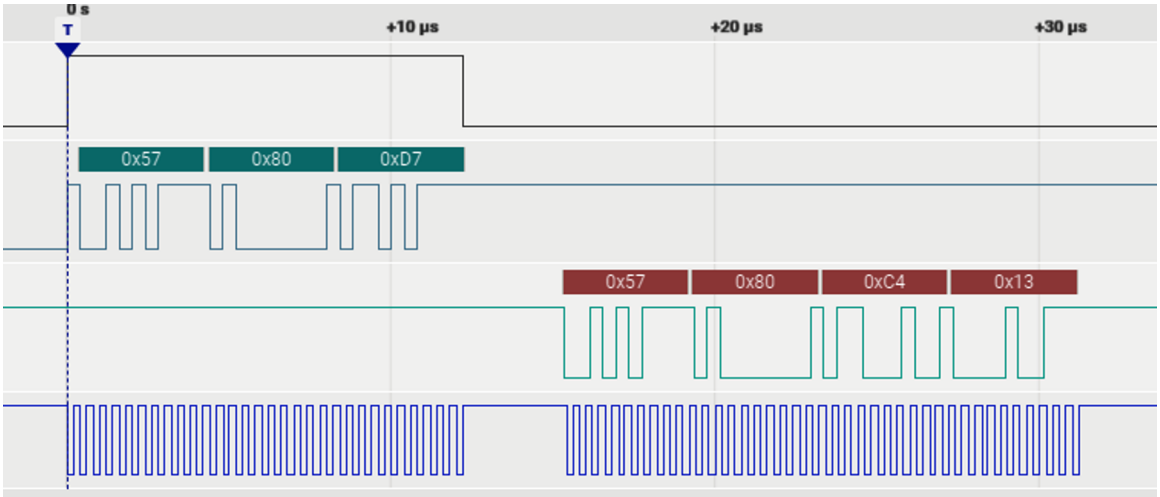


Figure 3-8. Data ID D Waveform

Note

Only the F2837xD requires an external connection between the CLB's generated SPI clock and the SPICLK pin. Other devices have an internal connection between the CLB and the SPICLK. For devices with an internal connection, the SPICLK can also be brought out to a pin for monitoring. The test connection for SPI CLK is shown in the device input/output diagrams inSection 2.3.5.2.

Examine the waveforms:

1. The SPI CLK frequency is 2.5 MHz as required by the T-Format specification.
2. The ControlField in the request matches the ControlField in the response. For example, in Data ID2, the ControlField is 0x49. The example software checks for this match and will halt if there is an error.
3. The number of fields in the response is expected based on the request.
4. The CRCField (last field) in the response is correct for the data received. The example software checks for this and will halt if there is an error.
5. Verify the encoder ID based on your encoder's specification.
6. Try different cable lengths, up to 100 m, and observe the change in the waveforms. Only the time between the request and response should change.

Manually turn the motor or encoder's shaft:

1. Turn the shaft in one direction. Notice the position and turns in the watch-window changes.
2. Turn the shaft in the opposite direction. Observe the position and turns change in the opposite direction.

3.3.4 Cable Length Validation

Table 3-4 lists tests with various types of encoders; cable length tests are performed. Tests include basic command-set exercising and reading-position values, with additional data if applicable.

Table 3-4. Cable Length Test Report

ENCODER NAME	TYPE	RESOLUTION (BITS)	CABLE LENGTH(1) (m)	MAXIMUM T-FORMAT CLOCK	TEST RESULT
TS5702N40	Rotary	17 bits	70 m	2.5 MBPS	Pass
TS5700N8501	Rotary	24 bits	70 m	2.5 MBPS	Pass

(1) Cable lengths up to 100 m have also been tested with some of the encoders.

3.3.5 Benchmarks

Table 3-5 lists the C28x CPU cycles required to execute T-Format library functions from RAM. This data was collected using:

- C2000 Codegen Tools V22.6.0.LTS
- float_support: fpu32
- tmu_support: tmu0
- fp_mode: relaxed
- abi: eabi

Table 3-5. Cycle-Count Benchmarks

		Cycles: -O2 -mf2	
Function	**CRC Calculation(1)**	**TX CRC by VCRC(2) RX CRC by CLB(3)**	**TX and RX CRC by C28x Lookup Table(4)**
setupCommandReadEEPROM	Transmit	266	264
setupCommandWriteEEPROM	Transmit	280	291
setupCommandReadoutOrReset	N/A	231	222
spiRxISR	N/A	418	418
startOperation	N/A	19	14
receiveDataID0_1_7_8_C	Receive	41	118
receiveDataID2	Receive	35	81
receiveDataID3	Receive	64	218
receiveDataID6	Receive	57	101
receiveDataIDC	Receive	53	97
updateEncoderID	N/A	1	1
updatePositionsOrTurns	N/A	18	18

(1) The indicated functions calculate either a transmit-data CRC or a received-data CRC. N/A indicates the function does not require a CRC calculation.
(2) The VCRC extension to the C28x CPU calculates the transmit-data CRC.
(3) The CLB calculates the received-data CRC as the response is incoming. This method uses an additional CLB tile and requires CLB Type 2 or later.

(4) Both transmit and receive CRCs are calculated by the C28x using a look-up table.

Table 3-6 lists the code-size, in 16-bit words, corresponding to each of the library source files. The C28x lookup-table takes 256 words of RAM or Flash that is not reflected in this table.

Table 3-6. Code-size in 16-bit Words

	Code-Size: -O2 -mf2	
Source File	**TX CRC: VCRC RX CRC: CLB**	**TX and RX CRC: C28x Lookup Table**
pm_tformat_source	936	923
clb_config	480	246
pm_tformat_crc	26	148
pm_tformat_crc_vcrc	14	N/A

3.3.6 Troubleshooting

Examining the following waveforms can assist in troubleshooting. Refer to the I/O diagrams in the design description:

- The CLB generated SPI clock.
- The output data from the SPI. This is the request to the encoder.
- The input data to the SPI. This is the response.
- The TxEN signal. This signal must be high during the request transmission.
- The encoder data signal (D+/D-) between the RS485 line driver and the encoder. Note: The data is a differential signal. Therefore, observation requires a special probe.

1. If the request is not transmitted by the SPI:
   - Check the connection between the SPICLK and the CLB. The CLB drives the SPI CLK. This connection can be internal to the MCU on all devices except F2837xD. On F2837xD this connection must be made externally.
   - Observe the SPI registers after the command setup and again after start operation. If the SPI receives the clock, the TX FIFO level decreases while the RX FIFO level increases.
2. The encoder does not respond:
   - Confirm TxEN is high during the request transmission.
   - Check the SPI clock frequency. The clock frequency must be 2.5 MHz for the encoder to respond. If the clock is not 2.5 MHz, then check the frequency configuration (TFORMAT_FREQ_DIVIDER) in the system example header file.
   - Check that the encoder is properly powered. In some cases the LaunchPad is not able to supply the current required by the encoder. Try providing power to the encoder separately, making sure all grounds are tied together.
3. The SPI CLK is observed during transmit but not during the response time:
   - Check the encoder's connection to the test hardware.
   - If you have modified the design:
     - Check that the response pin is routed into the correct CLB tile and to the correct input of that tile. The CLB must detect the encoder's response. Only then does the CLB generate the CLB_SPI_CLK.
     - If the CLB is driving SPI CLK internally, check that the tile and tile output enable are correct. Only certain tiles can access a specific SPI module. This can change on different devices.

4 Design Files

To download the design files, see the product page at TIDM-1011.

5 Related Documentation

Getting Started:

1. Texas Instruments, *C2000 T-Format Encoder Interface Software Guide* (html, pdf).
2. Texas Instruments, *C2000 Academy*, delivers easy-to-use training modules that span a wide range of topics for all C2000 devices.
3. Texas Instruments, C2000 Academy: CLB Module
4. Texas Instruments, *C2000 Software Guide* includes an overview of C2000 software, software development kits and development tools.
5. Texas Instruments, *C2000 Real-Time Control MCU Peripherals Reference Guide* The *CLB Type* indicates the feature set found on a specific implementation. The CLB type information for a particular C2000 MCU is indicated in the device data sheet and in this Reference Guide.
6. Texas Instruments, *CRC Engines in C2000 Devices*

Reference Designs:

1. Texas Instruments, *DesignDRIVE Development Kit IDDK v2.2.1 - User's Guide*
2. Texas Instruments, *DesignDRIVE Development Kit IDDK v2.2.1 - Hardware Reference Guide*
3. Texas Instruments, *C2000 DesignDRIVE*, software for Industrial Drives and Motor Control
4. Texas Instruments, *C2000 Position Manager SinCos Library*, User's Guide

5.1 Trademarks

C2000™, BoosterPack™, and TI E2E™ are trademarks of Texas Instruments.
LaunchPad™ is a trademark of Texas Instruments Incorporated.
All trademarks are the property of their respective owners.

6 Terminology

ABSx	From the T-format specification. ABS0:ABS1:ABS2 is the absolute position data for 1 revolution
ABMx	From the T-format specification. ABM0:ABM1:ABM2 is the multi-turn data.
ADF	From the T-format specification. The address data field used in read/write of EEPROM.
ALMC	From the T-format specification. Encoder error field.
BOOSTXL-POSMGR	See Position Manager BoosterPack
C28x	Refers to devices with the C28x CPU core
CF	From the T-format specification. ControlField. First field in any request and any response.
CLB	Configurable logic block
Command (or request)	Sent from the encoder interface to the encoder. The command determines what information is sent back from the encoder.
CPLD	Complex Programmable Logic Device
CRC	Cyclic redundancy check. T-Format uses the polynomial X^8 + 1.
Data ID code	From the T-format specification. 4-bit code + parity that identifies a specific T-Format request.
Delimiter	From the T-format specification. 1 bit, always 1, at the end of each field.
DFx	From the T-format specification. DataField within the response. The fields used and their content depend on the request.
EDF	From the T-format specification. EEPROM data field. Used for EEPROM read/write.

Encoder Interface	Logic which provides a controller, such as a C2000 Real-Time MCU, an interface to the communication protocol of an absolute encoder.
ENID	From the T-format specification. Encoder ID.
Field	From the T-format specification. Any request, or response, is made-up of one or more 10-bit fields.
FPGA	Field Programmable Gate Array
PM_tformat	Prefix used for all the encoder interface reference implementation functions. PM stands for position manager.
Position Manager BoosterPack (BOOSTXL-POSMGR)	The TIDM-1011 board is identical to the C2000 Position Manager BoosterPack plug-in module (see Section 2.3.3)
Request	See command
Sink Code	From the T-format specification. Fixed pattern of 0,1,0 immediately after the start-bit in the ControlField (CF)
SF	From the T-format specification. StatusField.
SPI	Serial peripheral interface.
Start-bit	From the T-format specification. 1 bit, always 0, at the start of any T-Format field.
Subsequent Electronics	T-Format encoder interface + controller implementation
T-Format	A communication protocol specification used by absolute encoders made by Tamagawa

7 About the Authors

LORI HEUSTESS has been a member of the C2000 team for many years. Her areas of interest have been in CPU and peripheral validation, software development, and industrial applications. Lori currently works on the C2000 Industrial Applications team.

SUBRAHMANYA BHARATHI AKONDY has worked on the architecture definition and design of several C2000 MCU products and control peripherals. His interests include MCU architecture, applications, and design aspects.

SHEENA PATEL works on the Industrial Drives team of the C2000 MCU group as a Product Marketing Engineer.

8 Revision History

Changes from Revision D (October 2022) to Revision E (July 2023) **Page**

- Updated the numbering format for tables, figures, and cross-references throughout the document.... 1
- Added *LaunchPad for TMS320F28P65x* 1
- Added *TMS320F28P65x* support to Supported Devices and LaunchPads table.... 4
- Updated images for clarification and consistency.... 12
- Updated *Input / Output and Tile Summary per Device* table to include TMS320F28P65x.... 12
- Updated *F2838x Input and Output for TMDXIDDKF273XD* image to include TMS320F28P65x.... 12
- Added to the software guide content list.... 34
- Added data to the troubleshooting list.... 42

Changes from Revision C (September 2020) to Revision D (October 2022) **Page**

- Added *LaunchPad resources and Software Development Kit (SDK)* 1
- Updated *clock frequency up to* to *clock frequency of* 1
- Added LaunchPad platforms and information about the Configurable Logic Block.... 4
- Updated the location of the transaction details into the Implementation Details section.... 8
- Changed title from *PM T-Format Master Details* to *C2000 T-Format Encoder Interface Overview* 8
- Deleted GPIO numbers and SPI instance information from the figure.... 8
- Updated legacy terminology for SPI.... 8
- Updated the last column in TIDM-1011 Board and BOOSTXL-POSMGR Connectors table.... 9
- Added LaunchPad header usage information.... 9
- Updated information in the MCU Resource Requirements.... 11
- Moved to the MCU resource requirements section.... 11
- Updated to indicate SysConfig is used.... 11
- Added details to the CLB Tile Usage table and added a table title.... 17
- Changed the format of the state transition description paragraphs to an ordered list of steps and included additional details.... 17
- Added detail to table 2-2 (MCU Resource Requirements).... 17
- Updated the directory information into a table (was in paragraph list format).... 17
- Updated figures to clarify input sources, removed obsolete terminology, and added additional clarifications. Corrected the inputs to LUT0 in CLB Outputs – Clock to SPI figure.... 17
- Consolidated some details into the hardware and software chapters.... 30
- Updated LaunchPads supported and CCS version required.... 32
- Added SW architecture diagram and description.... 34
- Added content.... 36
- Added new content.... 36
- Added new content.... 36
- Updated tool versions and note regarding building CLB-based projects.... 37
- Updated directory information into single location.... 37
- Updated Step 2 correcting the connector names for LaunchPad site-2.... 38
- Added a note that an external power supply for the encoder might be required.... 38
- Added a table with the location of example projects and added detail to the instructions.... 40
- Added the C2000 Academy.... 43
- Updated the terminology list.... 43

Changes from Revision B (January 2020) to Revision C (September 2020) **Page**

- Updated the numbering format for tables, figures and cross-references throughout the document.... 2

Changes from Revision A (November 2019) to Revision B (January 2020) **Page**

- Added paragraph describing figures 6-11.... 17
- Added figures 6-11 to show a logic schematic lens.... 17

Changes from Revision * (April 2018) to Revision A (November 2019)	Page
• Added *Configuration of Inputs to CLB* section	11
• Added *Implementation Details* section	17
• Changed information in *Software* section	34
• Changed encoder number in *Cable Length Test Report* table	41

IMPORTANT NOTICE AND DISCLAIMER

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

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265