

TI Designs

4-Axis CNC Router With 250-kHz Control Loop Reference Design



Description

This TI Design implements an application example on the simple open real-time Ethernet (SORTE) protocol with the programmable real-time unit and industrial communication subsystem (PRU-ICSS). The application example is based on a SORTE master and device, which drives a stepper motor with a frequency of up to 250-kHz step frequency. SORTE protocol enables customer applications to exchange process data between the master and devices in a 4- μ s cycle time. The design contains open source PRU firmware to enable customer to differentiate the products.

Resources

TIDEP0061	Design Folder
AM3359	Product Folder
TLK110	Product Folder
DP83822I	Product Folder
DRV8711	Product Folder
TMDSICE3359	Tools Folder
BOOST-DRV8711	Tools Folder

Features

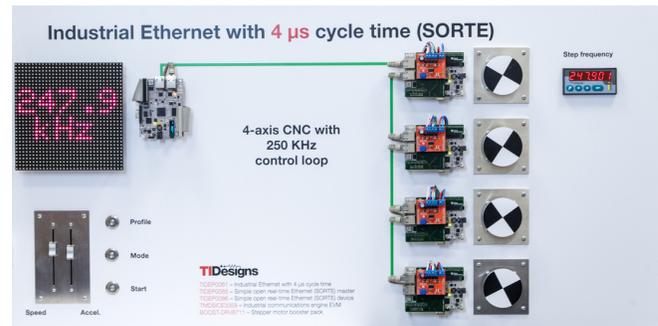
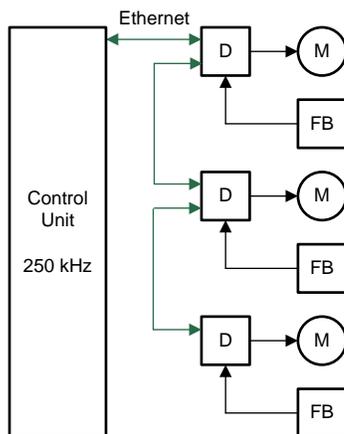
- Application Example Driving a Stepper Motor
- Based on SORTE Protocol
- Enables 4- μ s Cycle Time to Exchange Process Data
- PRU Firmware Provided in Source Code
- Fully-Customizable PRU Firmware
- Stepper Motor Application Example Supporting 250-kHz Stepper Frequency

Applications

- Industrial Ethernet
- [Motion Control](#)
- [Programmable Logic Controllers \(PLC\)](#)
- [Servo Drives](#)
- [Stepper Drives](#)
- [CNC Control](#)
- Textile Machines



[ASK Our E2E Experts](#)



An IMPORTANT NOTICE at the end of this TI reference design addresses authorized use, intellectual property matters and other important disclaimers and information.

1 System Overview

1.1 System Description

CNC routers are machinery used to mill out blocks of material. CNC routers use stepper motors to drive multiple axes. The axes of movement are in the directions of X, Y and Z. The fourth axis is used to tilt the CNC drill-head.

The control unit in a CNC router drives the stepper motors that are connected to each axis. Traditionally, the wiring between the control unit and each motor unit is implemented by two signals: a step (STEP) and a direction (DIR) signal.

Figure 1 shows the traditional CNC wiring between the control unit and the motor units.

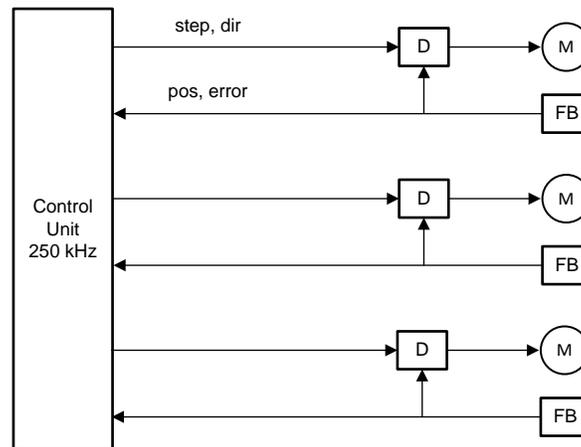


Figure 1. Traditional CNC Wiring Between Control Unit and Motor Units

A pulse on the STEP signal turns the stepper motor by a fixed angle. The amount of turn is determined by the stepper motor driver and the stepper motor itself. The DIR signal is used to drive the direction of the turns. The direction can be clockwise or counterclockwise.

The motor unit provides status and control information (for example, status information, motor overheating, and motor stall). This information is transmitted back to the control unit through a return channel. In many systems the return channels is proprietary protocol that is carried over serial or LVDS interface.

Traditional systems also use a star wire topology approach, which means that the STEP, DIR, and return channels route from the control unit to each motor unit.

This TI Design replaces the STEP, DIR, and return channels with industrial Ethernet based on the simple open real-time Ethernet (SORTE) protocol, which supports cycle time down to 4 μ s. The TI Design generates the STEP and DIR signal in the control unit and processes the STEP and DIR signals in the motor unit. Status and diagnostic data is sent over the industrial Ethernet return channel.

1.2 Key System Specifications

Table 1. Key System Specifications

PARAMETER	SPECIFICATIONS
250 kHz control loop	Master to device control loop cycle over industrial Ethernet SORTE protocol
DRV8711 booster pack voltage support	8- to 52-V supply input with up to 4.5-A continuous output current from each H-bridge
Micro-stepping support	DRV8711 supports built in 1/256-step micro-stepping indexer for ultra-smooth movement
4- μ s cycle time	Industrial Ethernet SORTE protocol between master and up to four devices
Automatic device detection	Automatic detection of devices at startup of industrial Ethernet SORTE protocol
Automatic device configuration	Automatic configuration of devices at startup of industrial Ethernet SORTE protocol
DRV8711 booster pack configuration	SPI interface for driver settings and status reporting

1.3 Block Diagram

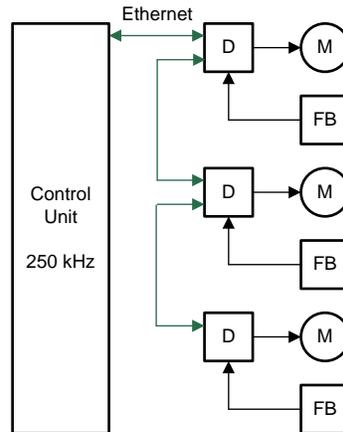


Figure 2. Block Diagram

1.4 Highlighted Products

1.4.1 AM3359

- Up to 1-GHz Sitara™ ARM® Cortex®-A8 32-bit RISC processor
- NEON™ SIMD coprocessor
- 32KB of L1 Instruction and 32KB of data cache with single-error detection (parity)
- 256KB of L2 cache with error correcting code (ECC)
- 176KB of on-chip boot ROM
- 64KB of dedicated RAM
- Emulation and debug - JTAG
- Interrupt controller (up to 128 interrupt requests)

PRU-ICSS:

- Supports protocols such as EtherCAT®, PROFIBUS, PROFINET®, EtherNet/IP, and more
- Two PRUs 32-bit load and store RISC processor capable of running at 200 MHz
- 8KB of instruction RAM with single-error detection (parity) 8KB of data RAM with single-error detection (parity)
- Single-cycle 32-bit multiplier with 64-bit accumulator
- Enhanced GPIO module provides shift-in or shift-out support and parallel latch on external signal
- 12KB of shared RAM with single-error detection (parity)
- Three 120-byte register banks accessible by each PRU INTC for handling system input events
- Local interconnect bus for connecting internal and external masters to the resources inside the PRU-ICSS
- Peripherals inside the PRU-ICSS:
 - One universal asynchronous receiver and transmitter (UART) port with flow control pins, supports up to 12 Mbps
 - One enhanced capture (eCAP) module
 - Two MII Ethernet ports that support industrial Ethernet, such as EtherCAT
 - One management data input and output (MDIO) port

On-chip memory (shared L3 RAM):

- 64KB of general-purpose on-chip memory controller (OCMC) RAM
- Accessible to all masters

External memory interfaces (EMIF):

- mDDR(LPDDR), DDR2, DDR3, and DDR3L controller:
 - mDDR: 200-MHz clock (400-MHz data rate)
 - DDR2: 266-MHz clock (532-MHz data rate)
 - DDR3: 400-MHz clock (800-MHz data rate)
 - DDR3L: 400-MHz clock (800-MHz data rate)
 - 16-bit data bus
 - 1GB of total addressable space
 - Supports one x16 or two x8 memory device configurations
- General-purpose memory controller (GPMC)
- Flexible 8-bit and 16-bit asynchronous memory interface with up to seven chip selects (NAND, NOR, Muxed-NOR, or SRAM)
- Uses BCH code to support 4-, 8-, or 16-bit ECC
- Uses hamming code to support 1-bit ECC

See the [AM335x Sitara Processors](#) data sheet for a complete list of features.

1.4.2 DRV8711

- Pulse width modulation (PWM) Microstepping motor driver:
 - Built-In 1/256-step microstepping indexer
 - Drives external N-channel MOSFETs
 - Optional STEP and DIR pins
 - Optional PWM control interface for DC motors
- Flexible decay modes, including automatic mixed decay mode
- Stall detection with optional BEMF output
- Highly-configurable SPI serial
- Interface internal reference and torque DAC
- 8- to 52-V operating supply voltage range
- Scalable output current
- Thermally enhanced surface-mount package
- 5-V regulator capable of 10-mA load
- Protection and diagnostic features:
 - Overcurrent protection (OCP)
 - Overtemperature shutdown (OTS)
 - Undervoltage lockout (UVLO)
 - Individual fault condition indication bits
 - Fault condition indication pin

See the [DRV8711 Stepper Motor Controller IC](#) data sheet for a complete list of features.

1.4.3 DP83822I

- IEEE 802.3u compliant: 100BASE-FX, 100BASE-TX and 10BASE-T
- MII, RMII, and RGMII MAC Interfaces
- Low-power single supply options:
 - 1.8-V average (AVD) < 120 mW
 - 3.3-V AVD < 220 mW
- ± 16 -kV HBM ESD Protection
- ± 8 -kV IEC 61000-4-2 ESD Protection
- Start of frame detect for IEEE 1588 time stamp
- Fast link-down timing
- Auto-crossover in force modes
- Operating temperature: -40 to 125°C
- I/O voltages: 3.3 V, 2.5 V and 1.8 V
- Power savings features:
 - Energy efficient Ethernet (EEE) IEEE 802.3az
 - Wake-on-LAN (WoL) support with magic packet detection
 - Programmable energy savings modes
- Cable diagnostics
- BIST
- Management data clock (MDC) and MDIO interface

See the [DP83822 Robust, Low Power 10/100 Mbps Ethernet Physical Layer Transceiver](#) data sheet for a complete list of features.

1.4.4 TMSICE3359 Industrial Communication Engine EVM

Hardware specification:

- AM3359 ARM Cortex-A8
- DDR3, NOR flash, and SPI flash
- Organize light-emitting diode (OLED) display
- TPS65910 power management 24-V power supply
- USB cable for JTAG interface and serial console

Software and tools:

- SYS/BIOS real-time operating system (OS)
- Starterware base port
- TI's Code Composer Studio™ (CCS) integrated development environment (IDE)
- Application stack for industrial communication protocols
- Sample industrial applications

Connectivity:

- PROFIBUS interface
- CANOpen
- EtherNet/IP
- PROFINET
- Sercos III
- Digital inputs and outputs (I/O)
- SPI
- UART
- JTAG

See the [TMSICE3359 tools folder](#) for complete list of features and design resources.

1.4.5 DRV8711 BoosterPack™

Complete stepper motor drive stage in a small form factor (1.75 in × 2.00 in):

- Supports 8.2 to 52 V and up to 4.5-A continuous for each H-Bridge
- 4x CSD88537ND Dual 60-V N-Channel NexFET Power MOSFETs (12.5 mΩ)
- Motor stall and device fault LED indicators Fully protected drive stage including overcurrent, overtemperature, undervoltage, and motor stall detect

See the [BOOST-DRV8711 tools folder](#) for a complete list of features and design resources.

2 System Design Theory

2.1 Stepper Motor Driver and Stepper Sotor

The AM3359 processor provides two signals to control the stepper motor: STEP and DIR signal. The STEP and DIR signals are transformed by the stepper motor driver DRV8711 into PWM motor control signals.

The DRV8711 is configured through an SPI interface from the AM3359 processor. The DRV8711 must be configured to work with a specific stepper motor in a specific microstep operation mode. The DRV8711 supports microstepping modes from 1/1 (full step), 1/2, and 1/4 down to 1/256.

The TI Design uses the stepper motor [ST6018M2008-B](#) from Nanotec™. The stepper motor turns its axis for 1.8° for each step. 200 steps are needed for one full turn of the axis when the DRV8711 motor driver is configured for 1/1 step. When the DRV8711 is configured for 1/256 microstepping then 51200 steps are needed for one full turn of the axis. Microstepping enables a very precise control of the motor angle.

2.2 SORTE Protocol

The SORTE protocol supports a 4-μs cycle time of process data exchange between one master and four devices. This is a high-performance exchange of process data that cannot be achieved by other industrial Ethernet protocols. With SORTE protocol it is possible to operate the motor control loop on the master, which then exchanges process data with the devices at a 250 kHz update rate.

Figure 3 shows the data exchange of SORTE protocol during IO exchange state. A process data frame is send by the SORTE master to the four SORTE devices, where each SORTE device extracts its individual process data. On the return path, each SORTE device sends data.

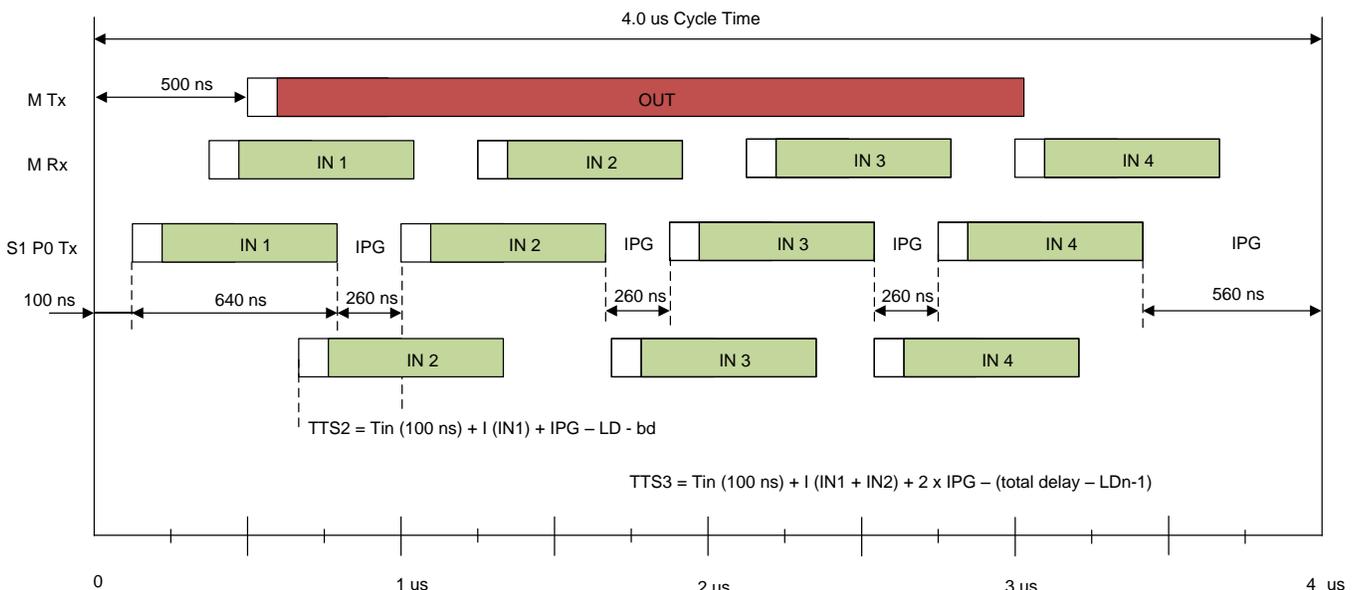


Figure 3. 4-μs Control Cycle Time of SORTE Protocol

SORTE protocol for the SORTE device requires auto-forward function of the MII_RT Ethernet hardware inside PRU-ICSS. The auto-forward function is available in devices that support all industrial protocols.

The SORTE protocol implementation on PRU-ICSS is described in TI Designs TIDEP0085[1] and TIDEP0086[2].

2.3 Control Unit (SORTE Master)

The control unit operates the SORTE master protocol. The master enters cyclic process data exchange state of SORTE protocol with all detected devices through the protocol startup sequence: discovery, parameterization, synchronization, and IO exchange.

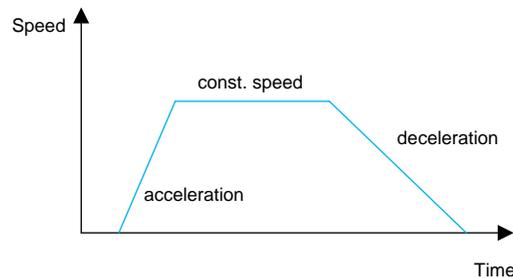


Figure 4. Speed Profile for Stepper Motor

The task of the control unit is to run the 250-kHz control loop. This loop includes the generation of the step profile including acceleration (step ramp-up), constant speed, and deceleration (ramp-down) of the stepper motor speed (see Figure 4). The master translates the generated step profile into STEP and DIR timestamp information and DRV8711 control commands. The master of SORTE protocol to the motor unit sends the timestamp information and control commands.

2.3.1 CNC Application From Master Side

The 4- μ s cycle time transfers step frequency and motor direction from SORTE master to four devices. Each device has a direct connection to the DRV8711 BoosterPack for stepper motor control. The maximum step frequency of 250 kHz ensures a minimum distance between two steps of 4- μ s. The DRV8711 stepper driver supports up to 256 microsteps. A complete turn, 200 steps \times 1.8 degree, with 256 microstepping takes 512000 microsteps.

Depending on the microstep setting and maximum step frequency, the motor will not support direct jump to step frequency but requires a ramp to approach maximum step frequency. A simple ramp is a linear increase in step frequency, which has its limits on start and the transition to full speed. The acceleration is smoother following a $\sin(x)^2$ curve as shown in Figure 5. The ramp is split into 32 intervals to go from zero speed to full speed, which is indicated by the 1 on the y-axis.

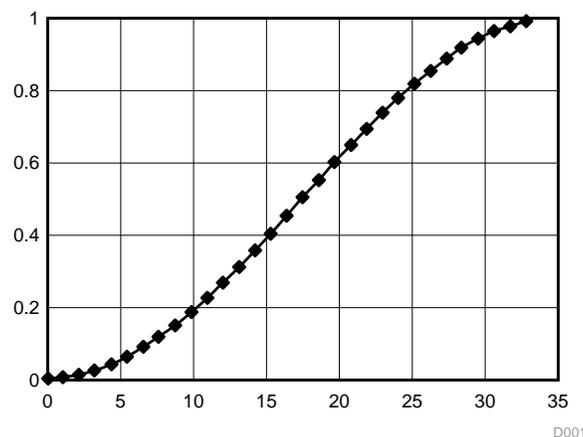


Figure 5. Normalized $\sin(x)^2$ Curve for Acceleration Ramp

For faster calculation of ramp data, the 32 values are stored in a look-up table, which is used for both acceleration and deacceleration of a motion. The actual step frequency is multiplied with the table entry for a given segment. PRU-ICSS hardware multiplier is used for this purpose.

2.3.2 Real-Time Calculation of Step Time Using PRU

The PRU core works on a 32-bit register and does not support floating point format on instructions. With 32-bit arithmetic the PRU core can either do floating point emulation using, for example, the C compiler in Code Composer Studio™ (CCS). With the hard real-time requirement to calculate new data and process packets every 4 μs, assembly coding is used. Following processing steps are required to calculate new step time.

1. Multiply the acceleration ramp entry with the step frequency to get the step time.
2. Divide the step time by 4 μs to split the time in number of cycles and step time less than 4 μs.
3. Add offset of the previous step time and adjust the cycle number if new step time exceeds 4 μs.
4. The new step time is split into 4us_cnt and sync_time for the next active cycle where the step is generated.

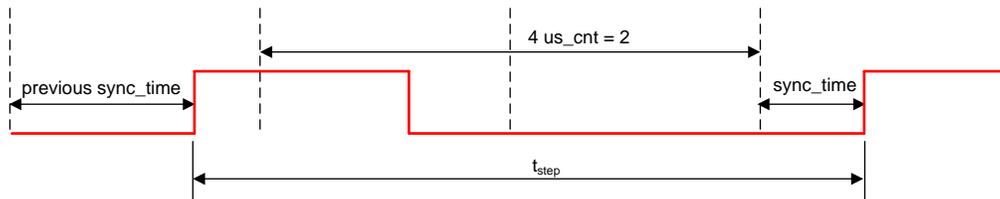


Figure 6. Calculation of New Sync Pulse

Equation 1 is the distance between two rising edges of a step pulse.

$$t_{step} (ns) = \frac{10^3}{\sin(x)^2} \times \frac{10^6}{f_{step}} \times 2^{16} \quad (1)$$

Equation 2 is the number of 4-μs periods before the next pulse.

$$4us_cnt = t_{step} \times \frac{2^{32}}{4000} \quad (2)$$

Equation 3 is the time of 4-μs wait periods.

$$t_{4us_cnt} (ns) = 4us_cnt \times 4000 \quad (3)$$

Equation 4 is the time of the sync pulse in the same cycle.

$$sync_time (ns) = previous_sync_time + t_{step} - t_{4us_cnt} \quad (4)$$

The equations above consider proper scaling to avoid fractions and divisions while maintaining enough precision of the result. The table entries for $\sin(x)^2$ ramp are stored $1/x$ times 1000. The constant for maximum step frequency is stored $1/x$ times 106 and multiplied with 216. Through upscaling of the operand for the first multiplication the result is in nanoseconds and shifted up by 16 bits. Equation 2 represents a division by 4000, which is solved by a multiplication with $1/4000$ and up-scaling of 32 bit. This is only possible as PRU 32-bit hardware multiplier supports 64-bit result.

The hardware multiplier uses register R25 to R29. R25 configures the multiplier for multiply only or multiply accumulate. XIN instruction with ID = 0 executes the multiplication in single cycle.

R28 (operand 1) * R29 (operand 2) is stored in R26 (lower word) and R27 (higher word).

Table 2. PRU Instruction Sequence

INSTRUCTION	R26	R27	R28	R29	COMMENT
—	—	—	$1/\sin(x)^2 * 1000$	$1/f_{step} * 10^6 * 2^{16}$	Load operands
XIN 0, &r26, 8	t_{step} low word in r26.w2	t_{step} high word in r27.w0	—	—	Multiply
—	—	—	t_{step}	$1/4000 * 2^{32}$	Load operands
XIN 0, &r26, 8	—	4_us_cnt	—	—	Multiply
—	—	—	4000	4us_cnt	Load operands
XIN 0, &r26, 8	t_{4us_cnt}	—	—	—	Multiply
SUB r29,r23,r26	—	—	—	sync_time	Subtract 4us_cnt

Table 2 shows the instruction sequence of three multiplications and one subtraction to get to the values of 4us_cnt and sync_time. The PRU takes 150 ns for the complete calculation of these two parameters, which include memory operations and boundary conditions for adjusting 4us_cnt in the case of previous sync_time plus current sync_time exceed 4 μs. In case the four motors should run completely independent, this calculation would need to be repeated. A total of 600-ns PRU calculation time for four independent axes still easily fits into the 4-μs cycle time. In this example, all four motors execute exactly the same sequence.

2.3.3 Control of CNC Application

The four motor CNC application firmware on PRU supports two modes. One mode is called position, and the other mode is called dance. In position mode the motors execute a given number of steps and a given step frequency. The motor can be selected as any single motor, all motors, or no motor. If no motor is selected, a special data pattern is transmitted to each motor to indicate that motor current can be turned off. These modes allow the user to manually adjust the motor into a reference position.

In dance mode the motors are running at a speed, which results in an acoustic tone. The sequence on tones is executed from a look-up table, which resides in PRU data memory. Each tone has three entries in the table: the tone frequency, the duration of the tone, and the pause to the next tone. As the number of steps is different for each tone when playing a ¼ note, the duration of a tone has different values.

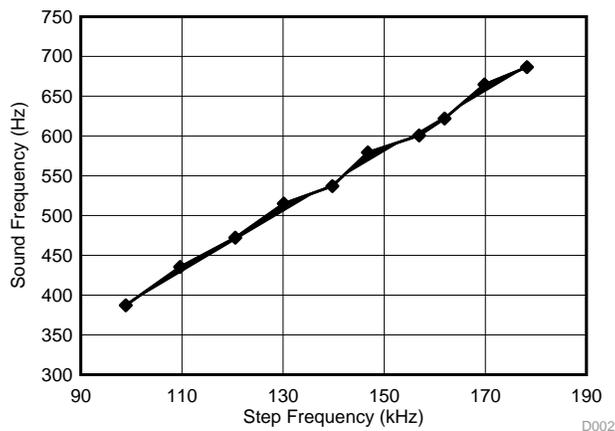


Figure 7. Sound Curve of Stepper Motors With Metal Plate

Figure 7 shows the measured acoustic tone with a given step frequency. The linear relation between the two frequencies was approximated by Equation 5.

$$x = (y - 9) \times \frac{49}{187} \tag{5}$$

Table 3. Step Frequency and Time for Acoustic Tones

tone	[Hz]	fstep [kHz]	1/f[Mhz]*2 ¹⁶	P125	P250
cis	277.1	70.25	932886	8781	17563
E	329.6	84.01	780121	10501	21002
fis	369.9	94.57	693009	11821	23642
G	391.9	100.33	653191	12542	25083
gis	415.3	106.46	615572	13308	26616
A	440.0	112.94	580294	14117	28234
H	493.8	127.03	515897	15879	31758
cis2	554.3	142.89	458659	17861	35722
D2	587.3	151.53	432486	18942	37883
fis2	739.9	191.52	342190	23940	47880

Based on the approximation, all tones required for the song were calculated and scaled for easy handling with PRU instructions. Columns P125 and P250 in [Table 3](#) represent the number of steps for each given tone at 125-ms and 250-ms duration. Each tone goes through the acceleration and deceleration curve.

The PRU firmware for the CNC application is in source file *cnc_app.asm*. The source file contains a function `CNC_APP_INIT`, which is called from *main.asm* during startup of the SORTe protocol. This initial function loads the $\sin(x)^2$ table and the tone table of the song into PRU data memory.

The SORTe master protocol calls the `CNC_APP_EXE` routine in the IO exchange state after sending the current output packet to four devices with motor attached. The current packet has the value for `sync_time` for each motor. This value gets directly loaded into SYNC generation of industrial Ethernet peripheral (IEP) timer on motor side. In case there is no pulse in the following 4- μ s cycle, a value of zero is sent to the motor and no SYNC pulse is generated.

The master side firmware has a state machine to define the transition from idle, ramp, continues speed, decline, and back to idle. Execution of the `sync_time` calculation starts at around 1.5 μ s after current output packet is handed over to MII TX fifo. The complete application takes less than 400 ns to run before the PRU goes into a polling mode to wait for the next 4- μ s boundary.

2.4 Motor Unit (SORTE Device)

2.4.1 TMSICE3359 to DRV8711 BoosterPack™ Interface Board

The interface board physically connects the SPI, SYNC0 and GPIO control signals between the TMSICE3359 and the DRV8711 BoosterPack. The following signals are used from the AM3359 processor on the TMSICE3359 board to control the DRV8711:

- SPI interface:
 - SPI1_SCLK - SPI instance 1, clock
 - SPI1_D0 - SPI instance 1, data out
 - SPI1_D1 - SPI instance 1, data in
 - SPI1_CS0 - SPI instance 1, chip select 0
- Control signals:
 - SYNC0 - STEP signal to DRV8711
 - GPIO1_31 - DIR signal to DRV8711
 - GPIO1_14 - RESET signal to DRV8711
 - GPIO1_15 - nSLEEP signal to DRV8711
 - GPIO0_12 - nSTALL signal from DRV8711
 - GPIO0_13 - nFAULT signal from DRV8711

NOTE: The male connector signal assignment for J3, J4, and J15 on the interface board are mirrored to fit the female connector layout type on the industrial communication engine (ICE) board.

Figure 8 shows the SORTE device hardware setup.

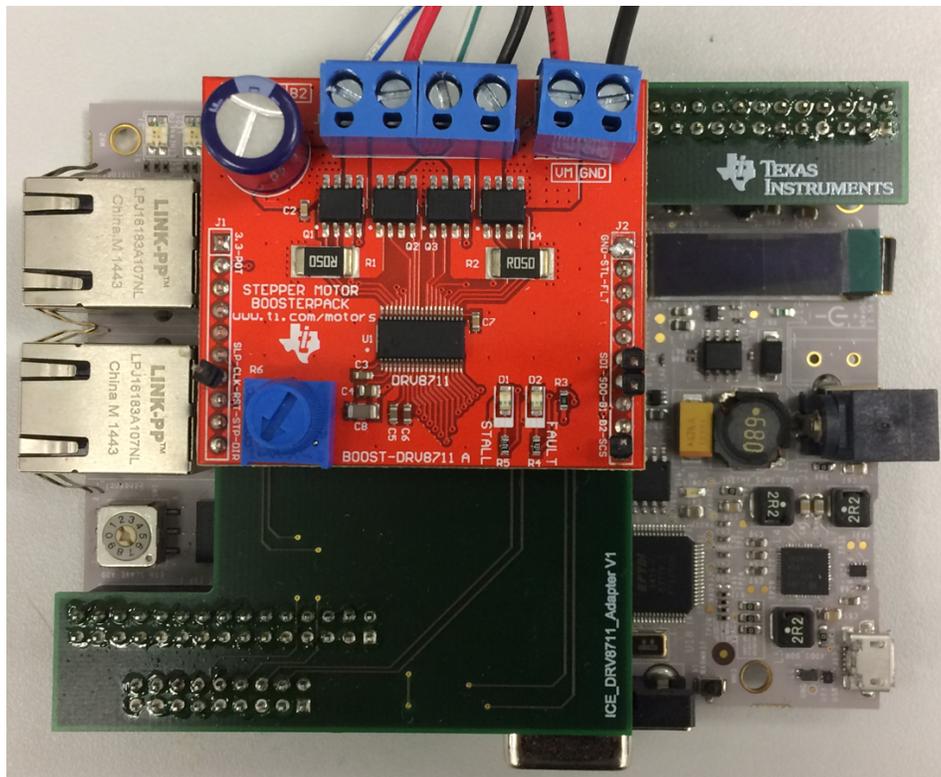


Figure 8. SORTE Device Hardware Setup

2.4.2 SYNC0 Pulse Generation

The IEP timer generates a programmable SYNC0 pulse, which is connected to the DRV8711 STEP signal input. The SYNC0 signal is a hardware-generated pulse and can be programmed in nanosecond granularity.

The PRU programs the pulse generation through setting a value in compare 1 register (CMP1). When the IEP timer matches with CMP1, a pulse on SYNC0 generates.

The SYNC0 pulse can be programmed with a start delay time through the START register. Programming a start delay means that when CMP1 has a match with IEP timer, the SYNC pulse gets delayed by the START register value. The SYNC0 pulse length is also programmed through the IEP registers. A SYNC0 pulse length of 2 μ s has been programmed to fulfill the minimum STEP pulse-length requirement of the DRV8711 device.

The SYNC0 signal generation is handled by the PRU firmware the following way:

1. The ioex.asm firmware module receives 4 bytes of the master process data. The process data is stored in PRU shared memory. 2 bytes of the process data represents the next SYNC0 pulse start time (referred to as PTO timer value) to get programmed in the next cycle period. The first 2 bytes are interpreted as follows:
 - 0x0000 - No SYNC0 pulse is generated in the next cycle.
 - 1 to 3990 - A SYNC pulse is generated in the next cycle. The SYNC0 start time is represented by the 2 bytes.
 - 0x7FFF - The DRV8711 is getting disabled by the ARM driver to save power consumption. Any other value than 0x7FFF enables the DRV8711.
 - 0x8000 (MSB of 2 byte value) - This data sets the direction signal of the DRV8711 to 1. If MSB is 0, then the direction signal is set to 0. This bit is used to control the rotation direction of the stepper motor.

The second 2 bytes are used by the master for sending a command to the device. Currently these 2 bytes are reserved for future use.
2. IEP timer events CMP0, CMP6, and CMP7 are used by PRU firmware to program the SYNC0 pulse. Each of the CMP events trigger a specific action in the PRU firmware and control the SYNC generation state machine.
 - CMP7: This event is set by PRU firmware to a fixed time offset within the 4- μ s cycle time. The CMP7 event starts a new SYNC0 programming cycle of the PTO timer value. Depending on the SYNC generation state machine, the pulse is set immediately by the CMP7 event handler or the setting of the pulse is delayed to CMP0 or CMP6 event.
 - CMP6: This event occurs after the SYNC0 generation has been occurred. The CMP6 event indicates that the SYNC0 has been completed and that the SYNC generation state machine can program the next SYNC0 pulse if pending.
 - CMP0: This event indicates the IEP timer wrap-around. The CMP0 event is used by the SYNC generation state machine to enable the CMP6 event and to program the next SYNC0 pulse.
3. SYNC generation state machine uses the following bit flags:
 - PENDING_PULSE_FLAG - A SYNC0 pulse has been programmed in CMP1, and the SYNC0 pulse is not yet completed.
 - ACTIVE_PULSE_FLAG - A CMP7 event will set this flag. The flag is cleared once the SYNC generation state machine is able to program CMP1 for the next SYNC0 pulse.
 - DELAYED_CMP6_ACTIVATED_FLAG - This flag is set if the SYNC0 pulse has a cycle time overlap (that is, the SYNC0 pulse is expanded over two cycles). The flag indicates to the SYNC generation state machine that CMP6 must be enabled with the next timer wrap around, which occurs with the CMP0 event.
 - IEP_CMP0_FLAG - The SORTIE IEP timer event handler sets this flag, which indicates to the SYNC generation state machine that the CMP0 event has occurred.
 - IEP_CMP6_FLAG - The SORTIE IEP timer event handler sets this flag, which indicates to the SYNC generation state machine that the CMP6 event has occurred.
 - IEP_CMP7_FLAG - The SORTIE IEP timer event handler sets this flag, which indicates to the SYNC generation state machine that the CMP7 event has occurred.

- **SYNC_PULSE_OVERLAP** - The SYNC generation state machine sets this flag when the generated SYNC0 pulse has a cycle time overlap. This information is used by the CMP0 and CMP6 event.
- **PREV_SYNC_PULSE_OVERLAP** - The SYNC generation state machine sets this flag when the previous SYNC0 pulse has a cycle time overlap. This information is used by the CMP0 and CMP6 event.

2.4.3 SYNC0 Pulse Generation Scenarios

The SYNC generation state machine programs the next SYNC0 pulse depending on the previous generation of SYNC0 pulses. The history of the last pulse is stored in a flag in PRU register memory. The following scenarios are supported by the SYNC0 generation state machine.

2.4.3.1 SYNC0 Pulse With No Pulse Generated in Previous Cycle Time

This scenario occurs when the SYNC generation state machine did not generate a SYNC0 pulse in the previous cycle time. The next SYNC0 generates as follows (see [Figure 9](#)):

- CMP1 value is set to 3990 (e/o cycle time), and CMP1 event is enabled.
- START value in IEP SYNC register is set to the delay time between 3990 and the actual SYNC0 pulse generation in the next cycle.
- The flag PENDING_PULSE_FLAG is set.
- CMP6 value is set to the actual SYNC0 pulse generation time plus 2- μ s. If the SYNC0 pulse has a cycle time overlap, the SYNC generation state machine sets the SYNC_PULSE_OVERLAP and DELAYED_CMP6_ACTIVATED_FLAG flags. CMP6 event is not enabled because this event will only occur in the next cycle time (or the next after cycle time if SYNC_PULSE_OVERLAP was set).
- CMP0 enables CMP6 if flag DELAYED_CMP6_ACTIVATED_FLAG is not set. Otherwise, the flag DELAYED_CMP6_ACTIVATED_FLAG clears.
- When CMP6 event occurs, the flag PENDING_PULSE_FLAG clears. This event enables the SYNC generation state machine to program the next PTO timer value.

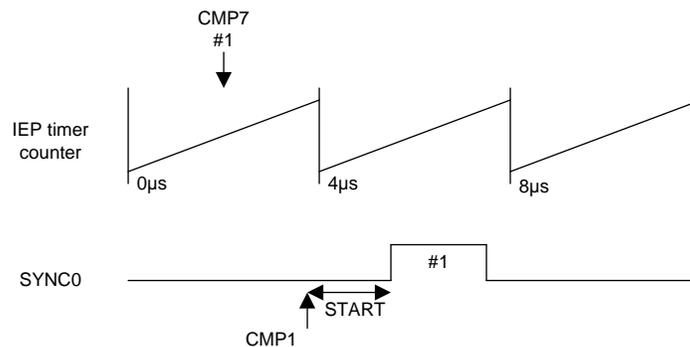


Figure 9. SYNC0 Generation With No Pulse Generated in Previous Cycle Time

2.4.3.2 SYNC0 Pulse Generation With No-Overlap Pulse Generated in Previous Cycle

This scenario occurs when a new SYNC0 pulse must be generated by the SYNC generation state machine, but the CMP6 has not yet indicated that the previous SYNC0 pulse is complete. This scenario also occurs when the previous SYNC0 pulse had no cycle time overlap. In this case the SYNC generation state machine delays programming the next SYNC0 pulse to the CMP0 event. Figure 10 shows this scenario.

- The #1 SYNC pulse is programmed by CMP7 #1 event. As the first SYNC0 has no pulse in the cycle time before, the pulse is programmed according to previous Section 2.4.3.1.
- When the CMP7 #2 occurs, the previous SYNC0 pulse is still active, CMP6 event has not yet occurred, and the flag PENDING_PULSE_FLAG is still set. The SYNC generation state machine delays the programming of the next SYNC0 pulse to when the next CMP0 event occurs.
- The following CMP7 #3 and #4 events follow the same programming approach as all previous pulses because the events did not have a cycle time overlap.

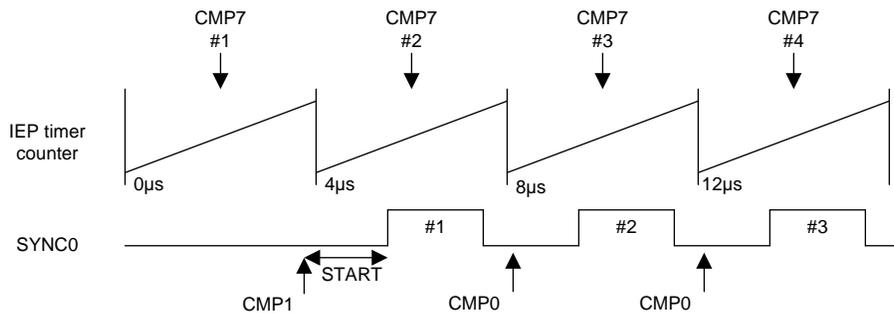


Figure 10. SYNC0 Pulse Generation With No-Overlap Pulse Generated in Previous Cycle

2.4.3.3 SYNC0 Pulse Generation With Overlap Pulse Generated in Previous Cycle

This scenario is when the SYNC generation state machine must generate a new SYNC0 pulse but CMP6 has not yet indicated that the previous SYNC0 pulse completed. This scenario is different from Section 2.4.3.1 because the previous SYNC0 pulse had a cycle time overlap.

- The #1 SYNC pulse is programmed by CMP7 #1 event. As the first SYNC0 pulse has no pulse in the cycle time before, SYNC0 is programmed according to Section 2.4.3.1
- The CMP7 #2 event cannot be programmed because the #1 SYNC0 pulse is still active. The #1 SYNC0 pulse also has a cycle time overlap.
- The SYNC generator state machine uses CMP6 to program the next SYNC cycle time.

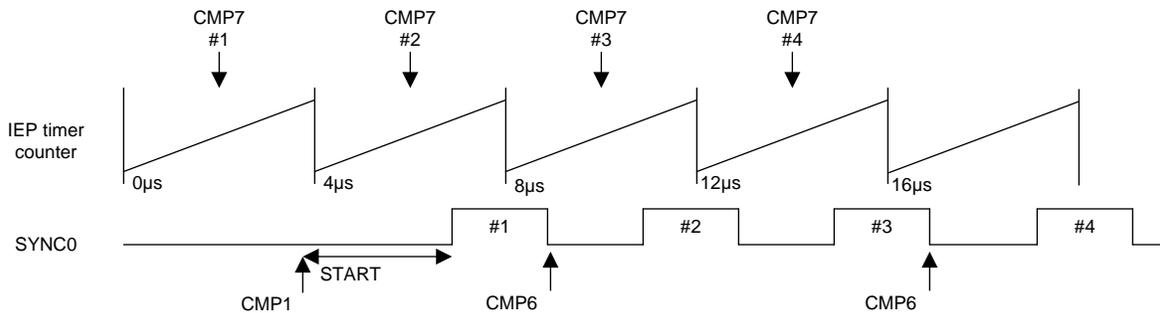


Figure 11. SYNC0 Pulse Generation With Overlap Pulse Generated in Previous Cycle

2.5 ARM® Application Software Description

The control unit (SORTE master) and motor unit (SORTE device) use the same ARM application source code. The ARM application reads out the rotary switch on the TMDSICE3359 to determine if this EVM operates as SORTE master or SORTE device.

The ARM application performs the following tasks:

- Configuration of AM3359 pinmuxing for industrial Ethernet ports of PRU-ICSS
- MMU configuration
- PRU-ICSS and INTC initialization
- PRU shared memory initialization and configuration
- CRC8 lookup table generation
- Loading and starting PRU firmware
- Master specific items
 - User interface
- Device specific items
 - DRV8711 device initialization
 - DRV8711 monitoring and power safe

3 Getting Started Hardware and Software

3.1 Hardware

The following boards are required for this TI Design:

- Between two and five [TMDSICE3359 ICE EVMs](#), available from the TI e-store.
- [DRV8711 BoosterPack EVMs](#) from the TI e-store, one for each connected stepper motor.
- one to four stepper motor ST6018M2008-B devices from Nanotec
- one to four interface boards to connect the TMDSICE3359 EVM to DRV8711 BoosterPack EVM (Gerber and Altium files provided with this TI Design)

For the control unit (SORTE master), the rotary switch on the TMDSICE3359 EVM must be set to position 0. The application software will then load the appropriate SORTE master PRU firmware and start the control unit application.

For the motor unit (SORTE slave), the rotary switch on the TMDSICE3359 EVM must be set to any other position aside from 0. The application software will then load the appropriate SORTE slave protocol and start the motor unit.

Connect the adapter board to connectors J3, J4, and J14 of the TMDSICE3359 board. Connect the DRV8711 BoosterPack to connectors J1 and J2. Connect the stepper motor according to the wiring instructions of the stepper motor and the DRV8711 BoosterPack. Power the TMDSICE3359 with 24 V and the DRV8711 BoosterPack with a 40-V power supply.

Connect an Ethernet cable from the master's RJ45 J2 port to device's RJ45 J2 port. With more than one device, connect an Ethernet cable from the first device's RJ45 J1 port to the second device's RJ45 J2 port. Follow the same procedure for the additional devices.

First, power up all devices before powering up the master. The master will detect the connected devices and will proceed to IO exchange state.

3.2 Software

Install the following software components on your development system:

- CCS version 6.1.3
- PRU compiler support for CCS
- Industrial SDK version 2.1.1.2
- SysBios version 6.42.2.29
- XDC Tools version 3.30.6.67_core

This TI Design is based on SORTE master and SORTE device software, available through TI Designs TIDEP0085[1] and TIDEP0086[2]. Download the software packages and import the software projects into CCS. There are three CCS projects:

- SORTE_app - ARM application
- SORTE_master - Master PRU firmware
- SORTE_slave - Device PRU firmware

Perform the following steps to compile the project and download the software to the EVM:

1. Copy the `appl_slave.asm` into the SORTE_slave project.
2. Copy the `cnc_app.asm` and `app_cnc_macros.inc` into the SORTE_master project.
3. Set the compiler flag `CNC_EXAMPLE` in the SORTE_master and SORTE_slave project.
4. Compile each project individually. Compile the SORTE_master and SORTE_slave project first as they each create a PRU firmware header file for the SORTE_app project.
5. Download the SORTE_app.out file through JTAG to each TMDSICE3359 EVM.
 - Alternatively, copy the file `SORTE_app_ti.app` to a bootable micro SD card, and rename the file to `app`. Insert the micro SD card into the micro SD card interface of the TMDSICE3359 EVM and press reset button or power cycle the board.
6. The application software is executing.

4 Testing and Results

4.1 Industrial Ethernet With 4- μ s Cycle Time

The setup consists of one SORTe master connected to four SORTe devices. Figure 12 shows the measurements on the first SORTe device of the Ethernet PHY with 4- μ s cycle time. Port 0 is connected through an Ethernet cable to the master, and Port 1 is connected towards the second devices. The network is in IO exchange state

- Port 0 RX_DV shows the reception of the master frame.
- Port 0 TX_EN shows the forward transmission of master frame.
- Port 1 RX_DV shows the received device frames. The three frames are forwarded to the master through PORT 0 TX_EN.

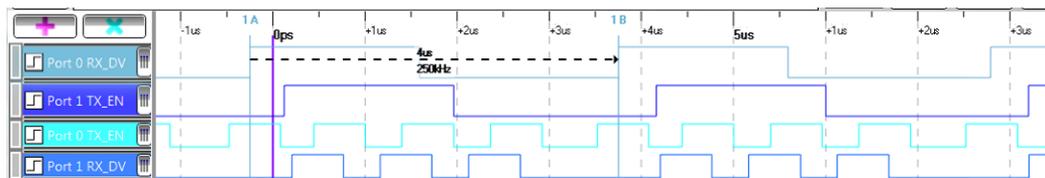


Figure 12. Industrial Ethernet With 4- μ s Cycle Time

4.2 DRV8711 Stepper Motor Current for 1/256 Microstepping

Figure 13 shows the measured stepper motor current for 1/256 microstepping with a motor stepper voltage of 40 V.

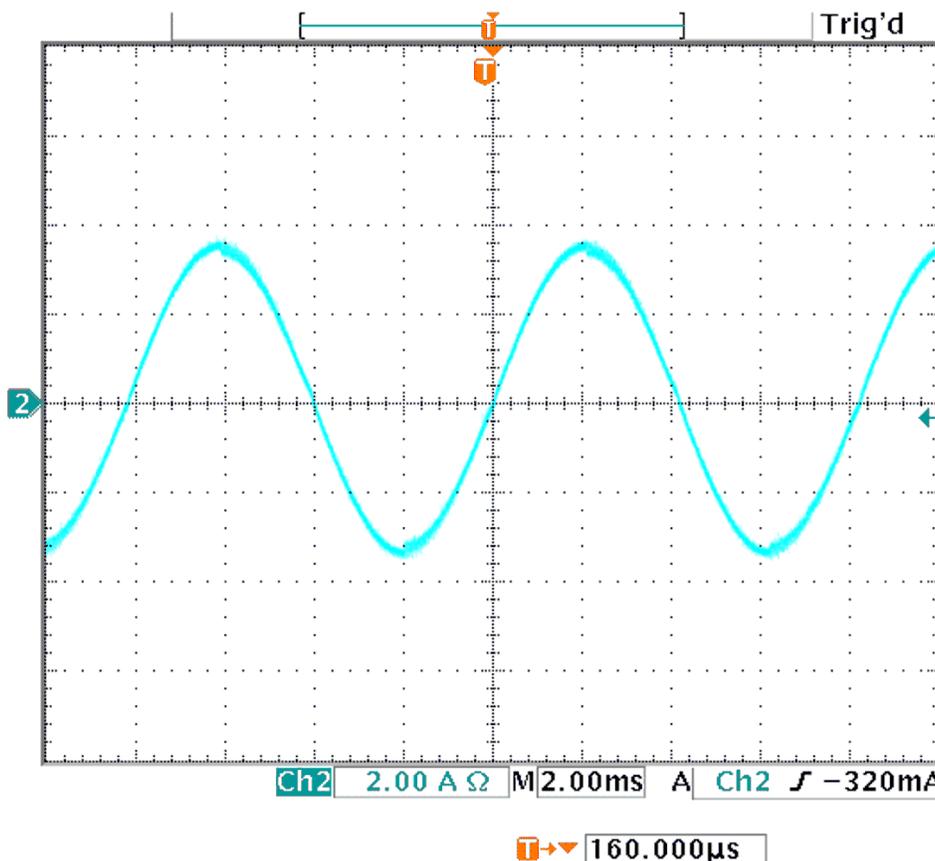


Figure 13. Stepper Motor Current for 1/256 Microstepping

5 Design Files

5.1 Schematics

To download the schematics, see the design files at [TIDEP0061](#).

5.2 Bill of Materials

To download the bill of materials (BOM), see the design files at [TIDEP0061](#).

5.3 PCB Layout Recommendations

5.3.1 Layout Prints

To download the layer plots, see the design files at [TIDEP0061](#).

5.4 Altium Project

To download the Altium project files, see the design files at [TIDEP0061](#).

5.5 Gerber Files

To download the Gerber files, see the design files at [TIDEP0061](#).

5.6 Assembly Drawings

To download the assembly drawings, see the design files at [TIDEP0061](#).

6 Software Files

To download the software files, see the design files at [TIDEP0061](#).

7 Related Documentation

1. Texas Instruments, [Simple Open Real-time Ethernet \(SORTE\) Master With PRU-ICSS Reference Design](#), TIDEP0085 TI Design (TIDUCK4)
2. Texas Instruments, [Simple Open Real-time Ethernet \(SORTE\) Device With PRU-ICSS Reference Design](#), TIDEP0086 TI Design (TIDUCK5)

7.1 Trademarks

All trademarks are the property of their respective owners.

8 Terminology

- CCS - Code Composer Studio
- ICSS - Industrial Communication System
- PRU - Programmable Real-Time Unit
- SORTE - Simple Open Real-Time Ethernet

9 About the Author

THOMAS MAUER is a System Engineer in the Factory Automation and Control Team at Texas Instruments Freising. He is responsible for developing reference design solutions for the industrial segment. Thomas brings his extensive experience in industrial communications like Industrial Ethernet and fieldbuses and industrial applications to this role. Thomas earned his degree in Electrical Engineering (Dipl. Ing. (FH)) at the University of Applied Sciences in Wiesbaden, Germany.

THOMAS LEYRER is a System Engineer in the Factory Automation and Control Team at Texas Instruments Freising.

SAGAR KUNDER is an Intern in the Factory Automation and Control Team at Texas Instruments, Freising. He is currently pursuing his Dual Master's Degree in Embedded Systems & Instrumentation from ESIGELEC, France and Manipal University, India. Sagar completed his Bachelor's Degree in Instrumentation Engineering from the University of Mumbai, India.

IMPORTANT NOTICE FOR TI DESIGN INFORMATION AND RESOURCES

Texas Instruments Incorporated ("TI") technical, application or other design advice, services or information, including, but not limited to, reference designs and materials relating to evaluation modules, (collectively, "TI Resources") are intended to assist designers who are developing applications that incorporate TI products; by downloading, accessing or using any particular TI Resource in any way, you (individually or, if you are acting on behalf of a company, your company) agree to use it solely for this purpose and subject to the terms of this Notice.

TI's provision of TI Resources does not expand or otherwise alter TI's applicable published warranties or warranty disclaimers for TI products, and no additional obligations or liabilities arise from TI providing such TI Resources. TI reserves the right to make corrections, enhancements, improvements and other changes to its TI Resources.

You understand and agree that you remain responsible for using your independent analysis, evaluation and judgment in designing your applications and that you have full and exclusive responsibility to assure the safety of your applications and compliance of your applications (and of all TI products used in or for your applications) with all applicable regulations, laws and other applicable requirements. You represent that, with respect to your applications, you have all the necessary expertise to create and implement safeguards that (1) anticipate dangerous consequences of failures, (2) monitor failures and their consequences, and (3) lessen the likelihood of failures that might cause harm and take appropriate actions. You agree that prior to using or distributing any applications that include TI products, you will thoroughly test such applications and the functionality of such TI products as used in such applications. TI has not conducted any testing other than that specifically described in the published documentation for a particular TI Resource.

You are authorized to use, copy and modify any individual TI Resource only in connection with the development of applications that include the TI product(s) identified in such TI Resource. NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY OTHER TI INTELLECTUAL PROPERTY RIGHT, AND NO LICENSE TO ANY TECHNOLOGY OR INTELLECTUAL PROPERTY RIGHT OF TI OR ANY THIRD PARTY IS GRANTED HEREIN, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information regarding or referencing third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of TI Resources may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

TI RESOURCES ARE PROVIDED "AS IS" AND WITH ALL FAULTS. TI DISCLAIMS ALL OTHER WARRANTIES OR REPRESENTATIONS, EXPRESS OR IMPLIED, REGARDING TI RESOURCES OR USE THEREOF, INCLUDING BUT NOT LIMITED TO ACCURACY OR COMPLETENESS, TITLE, ANY EPIDEMIC FAILURE WARRANTY AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

TI SHALL NOT BE LIABLE FOR AND SHALL NOT DEFEND OR INDEMNIFY YOU AGAINST ANY CLAIM, INCLUDING BUT NOT LIMITED TO ANY INFRINGEMENT CLAIM THAT RELATES TO OR IS BASED ON ANY COMBINATION OF PRODUCTS EVEN IF DESCRIBED IN TI RESOURCES OR OTHERWISE. IN NO EVENT SHALL TI BE LIABLE FOR ANY ACTUAL, DIRECT, SPECIAL, COLLATERAL, INDIRECT, PUNITIVE, INCIDENTAL, CONSEQUENTIAL OR EXEMPLARY DAMAGES IN CONNECTION WITH OR ARISING OUT OF TI RESOURCES OR USE THEREOF, AND REGARDLESS OF WHETHER TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

You agree to fully indemnify TI and its representatives against any damages, costs, losses, and/or liabilities arising out of your non-compliance with the terms and provisions of this Notice.

This Notice applies to TI Resources. Additional terms apply to the use and purchase of certain types of materials, TI products and services. These include; without limitation, TI's standard terms for semiconductor products (<http://www.ti.com/sc/docs/stdterms.htm>), [evaluation modules](#), and [samples](http://www.ti.com/sc/docs/sampterm.htm) (<http://www.ti.com/sc/docs/sampterm.htm>).

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2017, Texas Instruments Incorporated