Industrial Software Development Kit

Examples - AM335x and AM437x 2.1 Industrial Software Development Kit
## SDK Examples

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>EnDat Diagnostic</strong></td>
<td>Sample application demonstrating EnDat master using PRU-ICSS.</td>
</tr>
<tr>
<td><strong>EtherNet/IP</strong></td>
<td>Sample application demonstrating EtherNet/IP adapter using PRU-ICSS. This can be configured to operate as a single standalone switch.</td>
</tr>
<tr>
<td><strong>Ethernet MAC</strong></td>
<td>Sample application demonstrating a two port switch with two independent ethernet MACs with independent MAC and IP addresses using PRU-ICSS</td>
</tr>
<tr>
<td><strong>EtherCAT</strong></td>
<td>Sample application demonstrating EtherCAT slave functionality that can demonstrate both simple I/O and a motor control using PRU-ICSS</td>
</tr>
<tr>
<td><strong>Example Utils</strong></td>
<td>Sample application demonstrating different peripherals on the board.</td>
</tr>
<tr>
<td><strong>Motor control</strong></td>
<td>Sample application demonstrating motor control using PRU-ICSS.</td>
</tr>
<tr>
<td><strong>Profibus</strong></td>
<td>Sample application demonstrating a Profibus slave using PRU-ICSS.</td>
</tr>
<tr>
<td><strong>PROFINET RT/IRT</strong></td>
<td>Sample application demonstrating a PROFINET RT/IRT slave using PRU-ICSS.</td>
</tr>
<tr>
<td><strong>Sigma Delta Decimation Filter GUI client</strong></td>
<td>Sample application demonstrating Sigma delta A/D conversion using PRU-ICSS</td>
</tr>
</tbody>
</table>
Examples Agenda

- Overview
- EnDat
- Ethernet / IP
- Ethernet MAC
- EtherCAT
- Motor Control
- Example Utils
- Profibus DP
- PROFINET RT/IRT
- Sigma Delta Decimation Filter GUI client
- SDK Guides
Examples Overview

The following examples are included in the SDK with build configurations for am335x_release, am437x_release.

Supported Boards

<table>
<thead>
<tr>
<th>Application/Board</th>
<th>AM437x IDK</th>
<th>AM335xICEv2</th>
</tr>
</thead>
<tbody>
<tr>
<td>EnDat Master</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>Ethernet MAC</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>EtherNet/IP</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>EtherCAT</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Example Utils</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Motor Control</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>Profibus</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>ProfinetRT/IRT</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SDDF GUI client</td>
<td>✓  ***</td>
<td>✗</td>
</tr>
</tbody>
</table>

***Requires a separate Sigma Delta modulator and adapter card for AM437x IDK
EnDat Master Diagnostic

• The EnDat diagnostic application for SYS/BIOS demonstrates the EnDat master operation on the AM437x.

• EnDat is a bidirectional interface for position encoders. During EnDat operation the EnDat master requests and receives position other status information from the EnDat position encoder.

• This application is controlled with a terminal interface using a serial over USB connection between the PC host and the EVM and a serial terminal application (teraterm/ hyperterminal/ minicom).

• This example uses the EnDat driver which provides a defined set of API's to support the EnDat master interface.
  – The diagnostic invokes these API's to initialize the EnDat, configure the host trigger mode, select the channel, run the firmware, obtain and display the encoder information, compensate for propagation delay, and enables the user command interface.
  – The user may ether send an EnDat command to the encoder or configure the interface for motor control position feedback operation.

• For more information please refer to the SYSBIOS Industrial Software Development Kit EnDat Diagnostic Example Section, ENDAT System Reference Design and Reference Design for an Interface to a Position Encoder with EnDat 2.2

• The EnDat mode commands are shown on the next page.
# EnDat Master Commands

<table>
<thead>
<tr>
<th>Mode Command</th>
<th>EnDat Type</th>
<th>Mode</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encoder send position values</td>
<td>2.1</td>
<td>0</td>
<td>Request a position value</td>
</tr>
<tr>
<td>Encoder send position values with additional data</td>
<td>2.2</td>
<td>7</td>
<td>Request a position value and additional data, such as diagnostic, commutating, and acceleration in the same cycle</td>
</tr>
<tr>
<td>Selection of memory area</td>
<td>2.1</td>
<td>1</td>
<td>Select the memory area</td>
</tr>
<tr>
<td>Encoder send position values and selection of memory area or of the additional data 2</td>
<td>2.2</td>
<td>1</td>
<td>Request a position value and to select the memory area or block address in the same cycle</td>
</tr>
<tr>
<td>Encoder send parameter</td>
<td>2.1</td>
<td>4</td>
<td>Send parameters necessary for read access</td>
</tr>
<tr>
<td>Encoder send position values and send parameter</td>
<td>2.2</td>
<td>4</td>
<td>Request a position value and in the same cycle send parameters necessary for read access</td>
</tr>
<tr>
<td>Encoder receive parameter</td>
<td>2.1</td>
<td>3</td>
<td>Write parameters</td>
</tr>
<tr>
<td>Encoder send position values and receive parameter</td>
<td>2.2</td>
<td>3</td>
<td>Request a position value and write parameters in the same cycle:</td>
</tr>
<tr>
<td>Encoder receive reset</td>
<td>2.1</td>
<td>5</td>
<td>Execute a reset</td>
</tr>
<tr>
<td>Encoder send position values and receive error reset</td>
<td>2.2</td>
<td>5</td>
<td>Request position values and reset errors in the same cycle</td>
</tr>
<tr>
<td>Encoder receive test command</td>
<td>2.1</td>
<td>6</td>
<td>Write a test command</td>
</tr>
<tr>
<td>Encoder send position values and receive test command</td>
<td>2.2</td>
<td>6</td>
<td>Request position values and write a test command in the same cycle</td>
</tr>
<tr>
<td>Encoder send test values</td>
<td>2.1</td>
<td>2</td>
<td>Interrogate test values</td>
</tr>
<tr>
<td>Encoder receive communication command</td>
<td>2.2</td>
<td>2</td>
<td>Send communication data</td>
</tr>
</tbody>
</table>
**Ethernet / IP**

- Ethernet/IP an industrial networking standard that uses standard Ethernet and TCP/IP technologies and is compatible with convectional ethernet operations.

- EtherNet/IP is a member network family which use *Common Industrial Protocol* (CIP) at the application layer. CIP has a comprehensive suite of messages and services for numerous manufacturing automation applications, including control, safety, synchronization, motion, configuration and information.

- The EtherNet/IP adapter demo application uses a Molex EtherNet/IP stack running on top of NDK TCP/IP stack.

- The EtherNet/IP adapter application is implemented using the low latency cut-thru switch implementation on ICSS1.

- The PRU implements basic Ethernet switch protocols including features such as storm prevention and packet statistics.

- Industrial CommunicationsSubsystem (ICSS) architecture allows low latency store and forward between the ports based on configurable parameters.

- This example is limited to permit execution for one hour only. After that it terminates operation.

- This application uses the libsys_bios_driver.a, libboard_support.a and a few Starterware libs.
The Ethernet/IP on Industrial Communication SubSystem (ICSS) supports Beacon based DLR (Device Level Ring) topology (ring redundancy protocol specified by ODVA).

The main features of this implementation are:
- Supports 200 us beacon interval and 400 us beacon timeout interval
- Supports learning table exception for Supervisor
- Dynamic start and stop. User can enable or disable DLR on the fly
- No user configuration required. The ring parameters get configured on their own.

ACD (Address Conflict Detection) feature is enabled. Once the link is established, the IP Address is obtained once the Address Resolution Protocol probing is done and no duplicate IP Address has been detected.

For more information please refer to the SYSBIOS Industrial Software Development Kit Ethernet/IP Example Section and EtherNet/IP™ on Sitara Processors.

The application supports the following modes configured via the serial console:

<table>
<thead>
<tr>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start mode</td>
<td>Start stack in IO exchange mode</td>
</tr>
<tr>
<td>Stop mode</td>
<td>Stop IO exchange mode</td>
</tr>
<tr>
<td>Run mode</td>
<td>IO Data is exchanged with scanner</td>
</tr>
<tr>
<td>Idle mode</td>
<td>No Data exchange with scanner. Only communication.</td>
</tr>
<tr>
<td>Copy in-out mode</td>
<td>Copy the data from scanner to the OUT data</td>
</tr>
<tr>
<td>Increment mode</td>
<td>Stack will increment the OUT data periodically</td>
</tr>
<tr>
<td>Change IP Address</td>
<td>IP address of the device can be configured manually</td>
</tr>
<tr>
<td>Erase Non Volatile Memory</td>
<td>Erase the stored device configurations from the non volatile memory</td>
</tr>
</tbody>
</table>
Ethernet MAC

• This example is an Ethernet Dual mac implementation. The application has two MAC instances each having independent IP Address.

• This example runs on top of the ICSS EMAC driver and uses NDK as TCP/IP stack.

• ICSS EMAC Driver operates in emac mode so that both PRUs work independently.
  – The application can be modified to also run on a single MAC.

• The IP address used:
  – Port1 - 192.168.2.3 - IP Configuration of Instance 1 is assigned in the Stack init hook Callback
  – Port2 - 192.168.1.3 - IP address of Instance 2 is configured via CFG file

• Once the application is loaded the details are printed on the UART console. The TriColour LED will blink Green to indicate the application is running

• The Ethernet MAC application uses the libsys_bios_driver.a, libboard_support.a and a few Starterware libs.

• For more information please refer to the SYSBIOS Industrial Software Development Kit Ethernet MAC Example Section.
EtherCAT

- EtherCAT is a real-time low-latency Industrial field bus protocol that is becoming popular in industrial automation and motor control applications.

- The EtherCAT Industrial application, slave stack, Protocol Adaption Layer and PRU Subsystem API run on the ARM processor.

- The EtherCAT Layer 2 real-time functions are implemented in the ICSS.
  - These include datagram processing, distributed clocking, address mapping, error detection and handling, and host interface.
  - The PRUs also emulate EtherCAT register space in the internal shared memory.
EtherCAT

• Two versions are available
  – Simple Demo Application - An EtherCAT sample application built using pre-built EtherCAT stack library of the SDK.
    • Simple sample application interface with a 32-bit input and 32-bit output to toggle LEDs and read digital inputs
    • Object dictionary adaptation is not supported in this evaluation version.
  – Full Feature Demo Application - This application provides full configurability including motor control for the AM437x. However this application requires the Beckhoff Slave Sample Code (SSC 5.11) in source format from the Beckhoff website.
    • The Beckhoff TwinCat application provides the user interface.
    • The Motor Control demo demonstrates a 3 phase sensored Field Oriented Control (FOC) of a single Permanent Magnet Synchronous motor (PMSM).
      – using current feedback with the on chip AM437x ADCs or Sigma Delta Decimation Filtering
      – optionally using position feedback with EnDAT 2.2 position feedback.
    • The motor can be run open loop or in a closed (velocity or portion) loop, with EnDat position feedback.
EtherCAT

• This application uses the libsys_bios_driver.a, libboard_support.a, libecat_slave_stack_am437x.a and a few Starterware libs.

• For more information please refer to the
  
  – [SYSBIOS Industrial Software Development Kit](#) EtherCAT Example Section,
  
  – [EtherCAT on Sitara Processors](#)
  
  – [PRU ICSS EtherCAT Firmware API Guide](#)
  
  – [AM437x SingleChip Motor-Control Design Guide](#)
  
  – [AM437x Single Chip Motor Control Benchmark](#)
  
  – [Isolated Current Shunt and Voltage Measurement Reference Design for Motor Drives Using AM437x](#)
Motor Control

• The Motor Control demo demonstrates a 3 phase sensored Field Oriented Control (FOC) of a single Permanent Magnet Synchronous motor (PMSM)
  – Using current feedback with the on chip AM437x ADCs or Sigma Delta Decimation Filtering
  – Optionally using position feedback with EnDAT 2.2 position feedback.
  – The motor can be run in open or closed speed loop and in closed position loop, with EnDat position feedback

• User interface provided by a UART over USB interface using a serial terminal application (like teraterm/ hyperterminal/ minicom).

• This application uses the libsys_bios_driver.a, libboard_support.a, libecat_slave_stack_am437x.a and a few Starterware libs.
Motor Control

- The motor control example highlights the operation of a number of the motor control subsystems including:
  - ADC
    - ADC sampling firmware using ICSS0_PRU1
    - PRU ADC sampling firmware Host interface
    - ADC firmware initialization and operation sequence
  - Sigma Delta Decimation Filter
    - Overview
  - EnDat
    - Overview
  - Field Oriented Control (FOC)
    - Position Offset Compensation
    - PI Tuning

- For more information please refer to the SYSBIOS Industrial Software Development Kit Motor Control Example Section.
Example Utils

- **Example Utils** - The Example Utils application demonstrates the Digital Outputs, Digital Inputs, Rotary Switch and Flash memory (SPI on ICEv2 and QSPI on IDK) being accessed from a SYS/BIOS application.
  - This application also utilizes the LCD panel on ICEv2.
  - User Interface - UART terminal controls to trigger various peripherals.
  - This application is dependent on the libsys_bios_driver.a, libboard_support.a and a few Starterware libs.
    - Project files for these libraries are included in the SDK.
    - Upon modifying and rebuilding libraries, the application should be rebuilt as well.
  - For more information please refer to the SYSBIOS Industrial Software Development Kit Example Utilities Section.
Profibus DP

- **PROFIBUS DP** (Decentralised Peripherals) is used to operate sensors and actuators under the command of a Profibus DP master controller in production (factory) automation applications.

- This example application is a AM335x implementing a Profibus DP *slave* device on a distributed in multi-drop RS485 serial bus with a Profibus DP master running on a PC.
  - **ProfiTrace** (Version 2.6.1 or Later) from Procentec can be used to implement Profibus master on a windows machine. The master provides the user interface and control functions.
  - The 'ProfiCore Ultra' hardware is connected between the PC and the EVM running Profibus slave application.

- This application uses drivers.lib, am335x_platform.lib, sys_bios_driver.lib, utils.lib, platform.lib, system.lib, and profibus.lib. The project files for these libraries except for profibus.lib are included in the SDK.
Profibus DP

- There are three software components in the PROFIBUS solution on TI devices.
  - ICSS-PRU micro code that implements Fieldbus Data Link (FDL) functionality in the PRU,
  - The PROFIBUS-DP protocol that runs on the ARM MPU
  - End equipment Industrial application that runs on the ARM MPU.

- The ICSS-PRU subsystem implements real-time PROFIBUS (Layer 2) message transmission, frame validation and communication with the ARM processor.

- Interrupts are used to communicate with the ARM where the PROFIBUS stack (Layer 7, DP-Protocol) and the industrial application are run.

- Communication between the ARM and PRU are handled through the internal memory.

For more information please refer to the [SYSBIOS Industrial Software Development Kit PROFIBUS Example](#) and [PROFIBUS on TI’s Sitara Processors](#)
PROFINET RT/IRT

- PROFINET is a real-time Ethernet standard for high-speed, deterministic communications used in a wide range of industrial applications including factory automation, process automation and building automation.

- PROFINET I/O RT/IRT Device has an integrated two-port cut-through learning switch.
  - Switch handles the non real-time traffic
  - Interfaces with the PROFINET stack and TCP/IP (NDK) stack.
    - PROFINET Quality of Service (QoS) using four priority queues on host and port interfaces.
    - PROFINET Filter Database for multicast
The PROFINET slave implementation has three major software components.

- The microcode that implements Layer 2 functionality in the device’s ICSS-PRUs
- The PROFINET slave stack that runs on the ARM
- The industrial application that runs on the ARM.

TI provides additional components such as the protocol adaptation layer and device drivers in the SDK.

- The PRUs perform the tasks of CPM/PPM processing, Data Hold Timer (DHT), DCP Identify Filter, cut-through switching, error detection and host interface handling.
- The PRU also provide an easy-to-use PROFINET register space in the internal shared memory.
PROFINET RT/IRT

• A simple sample I/O Application has been provided to demonstrate the usage of PROFINET implementation.
  – The example is PROFINET I/O RT/IRT Device (slave) application based on Molex PROFINET stack.
  – Supports SNMP MIB-2(System and Interfaces), LLDP-MIB, LLDP-EXT3-MIB and LLDP-PNO-MIB which are mandated for Conformance Class B.
  – Current implementation of the LLDP-MIBs are limited in nature as the integration between PROFINET stack and SNMP interface is incomplete.
  – The SNMP stack available in the example is a limited version, and shuts down after 1024 SNMP requests.
  – For more information please refer to the SYSBIOS Industrial Software Development Kit PROFINET Example and PROFINET on TI's Sitara Processors
Sigma Delta Decimation Filter GUI client

• Sigma delta A/D converters combine oversampling analog sigma delta modulators with digital decimation filters to achieve high precision and cost effective A/D conversion.

• This application demonstrates Sigma delta A/D conversion using AM437x IDK along with AMC1304 Sigma Delta modulator.

• Default configuration is Sinc3 running a 20 MHz.
  – Firmware can also be configured to demonstrate a Sinc3 based over current detection capability.

• This application provides the sample value to host PC via UART.

• For more detail
  – To learn about on installing and using the GUI in host PC to observe and analyze the wave forms please see Isolated Current Shunt and Voltage Measurement Reference Design for Motor Drives Using AM437x.
  – For more information on the SDDF implementation please refer to the SYSBIOS Industrial Software Development Kit SDDF Example Section and Isolated Current Shunt and Voltage Measurement Reference Design for Motor Drives Using AM437x
SDK Guides

Building Full Feature EtherCAT Application
• The EtherCAT example application provided in the SDK is a limited development application.
  – To have a full development capability on AM335x/AM437x,
    • Get the EtherCAT source code from Beckhoff at [http://www.ethercat.org/memberarea/stack_code.aspx](http://www.ethercat.org/memberarea/stack_code.aspx) (for EtherCAT members)
    • Apply the patch file to modify Beckhoff source files for the AM335x/AM437x.
    • Rebuild the application

Building Full Feature SNMP stack
• SNMP is provided as a library in the Industrial SDK with a limit of 1000 SNMP requests (after which it will shutdown).
  – To have full development capability
    • Get the stack from Interniche SNMP Stack v4.01
    • Re-build the library to get full development access.
• For more information please refer to the SYSBIOS Industrial Software Development Kit.
SDK Guides

Building Bootloader

• Boot loader project is included in the starterware package.
• The supported build configurations for AM437x are:
  • Builds the binary to be executed from SD card
    – am43xx_boot_mmcisd_debug & am43xx_boot_mmcisd_release
    – am335x_boot_mmcisd_debug & am335x_boot_mmcisd_release
  • Builds the binary to be executed from qSPI flash
    – am43xx_boot_qspi_debug & am43xx_boot_qspi_release
    – am335x_boot_qspi_debug & am335x_boot_qspi_release
• Using the Starterware Bootloader project create the .out file.
• Use the OBJCOPY to generate a binary(.bin) from a .out file
• For more information please refer to the SYSBIOS Industrial Software Development Kit Building Bootloader section