参考資料

TI Designs
EtherCAT®マスター・リファレンス・デザイン、AM335x CPSW用

TEXAS INSTRUMENTS

TI Designs リファレンス・デザイン

このTI Designは、acontis社のEC-Masterスタックを使用して、Sitara™ AM335xプロセッサで動作する EtherCAT®マスター・インターフェイスの詳細を示します。このEtherCATマスター・ソリューションは、EtherCATベースのPLCやモーション制御アプリケーションに使用できます。EtherCATマスターは、AM335xプロセッサのCommon Platform Ethernet Switch (CPSW)ポート上でプロファイル作成されています。AM335xプラットフォームにこのEtherCATマスターを実装することで、100µs未満のサイクル時間を実現できます。

設計リソース

TIDEP0043	ツール・フォルダ
TMDSICE3359	ツール・フォルダ
TMDSIDK437x	ツール・フォルダ
AM3359	プロダクト・フォルダ

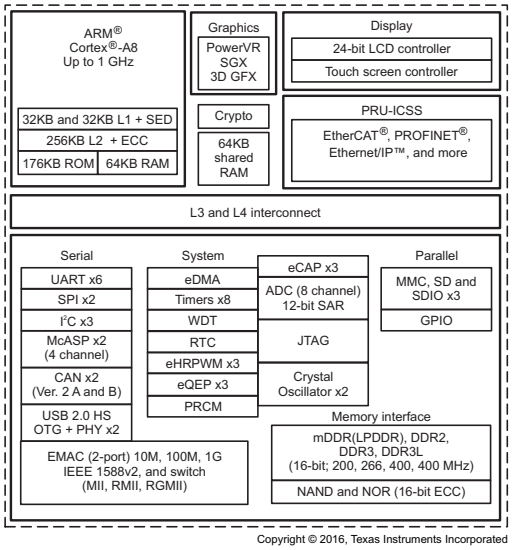
TI E2E™ Community　E2Eエキスパートに質問

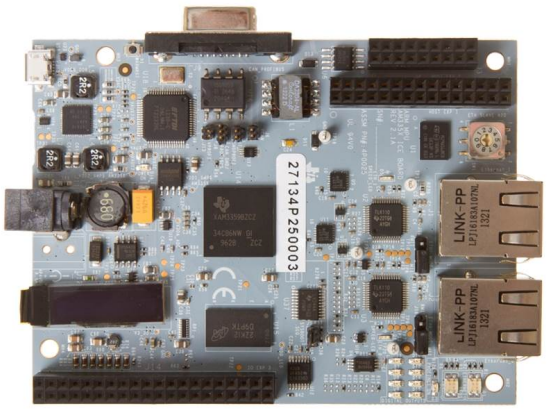
デザインの特長

- ETG.1500仕様に基づくEtherCAT Class AまたはClass Bのマスター・スタック
- 高性能CPSWイーサネット・ドライバで、EtherCATの最高性能を実現
- EtherCAT Feature Pack: Cable Redundancyで2個のCPSWポートまたは2個のICSS_PRUポートを使用
- EtherCAT Feature Pack: Hot Connectでフレキシブルな構成をサポート
- 各種オペレーティング・システムをサポート: Linux®、TI-RTOS (SYS/BIOS)、StarterWare、VxWorks®、および QNX™

主なアプリケーション

- EtherCATプログラマブル・ロジック・コントロール(PLC)システム
- EtherCATモーション・コントロール・アプリケーション
- EtherCATインターフェイス・ボード
- EtherCAT産業用通信ゲートウェイ





1 Introduction

This TI Design document presents the TI Sitara AM335x implementing an EtherCAT Master using acontis EtherCAT Master stack (EC-Master). The acontis EtherCAT Master stack is a highly portable software stack, and when combined with a high-performance TI Sitara CPU it provides a sophisticated EtherCAT solution that users can use to implement EtherCAT communication-interface boards, EtherCAT-based PLC, or EtherCAT-based motion control applications.

The EC-Master architectural design does not require additional tasks, making it easier to transport code to a different OS or to bare-metal systems. Due to this architecture combined with the high-speed Ethernet driver, it is possible to implement applications on AM335x platform with short cycle times less than 100 µsec.

2 EtherCAT® Protocol

EtherCAT is an IEEE 802.3 Ethernet-based fieldbus system. EtherCAT is a new standard in communication speed. Because EtherCAT is inexpensive to implement, the system can use fieldbus technology in applications which previously omitted fieldbus. EtherCAT is an open technology that is standardized within the International Electrotechnical Commission (IEC). The technology is supported and powered by the EtherCAT Technology Group (an international community of users and vendors). The protocol is suitable for both hard and soft real-time requirements in automation technology. A primary advantage of EtherCAT is that it supports automation applications that require short data-update times with low communication jitter and reduced hardware costs.

In the EtherCAT protocol, the EtherCAT Master sends a telegram that passes through each node. Each EtherCAT Slave device reads the data that is addressed to it as soon as the data is detected. Then, the slave device inserts the data into the frame as the frame moves downstream. The frame is delayed by hardware-propagation delay times. The last node in a segment (or branch) detects an open port and sends the message back to the master using the full-duplex feature of Ethernet technology. The EtherCAT Master is the only node within a segment that actively sends an EtherCAT frame. All other nodes forward the frames downstream. This capability permits the network to achieve over 90% of the available network bandwidth, prevents unpredictable delays, and guarantees real-time system response.

The EtherCAT protocol is optimized for process data transfer and is transported within the Ethernet frame by using a special Ethertype identifier (0x88A4). EtherCAT communications consist of several EtherCAT telegrams. Each telegram serves a specific memory area of the logical process image, up to 4GB. The data sequence is independent of the physical order of the Ethernet terminals in the network.

In addition to data exchanges between the EtherCAT Master and Slave, EtherCAT is also suitable for communication between controllers (master to master). Freely addressable network variables for process data and a variety of services for parameterization, diagnosis, programming, and remote control cover a range of requirements. The data interfaces for master-to-slave and master-to-master communication are identical.

Two mechanisms are available for slave-to-slave communication. The first option is to use upstream devices to quickly communicate to downstream devices within the same cycle. The other option is using the freely configurable slave-to-slave communication that runs through the master device. The second option requires two bus cycles (although not necessarily two control cycles).

3 Block Diagram

3.1 TI Sitara™ Overview

The Sitara AM335x processor is low-power device based on the ARM® Cortex®-A8 RISC. This system on a chip (SoC) features a broad range of integrated peripherals that are suitable for industrial applications. Sitara processors support multiple operating-frequency ranges from 300 MHz for simple applications and up to 1 GHz for more complex high-performance applications. The AM335x processor is configured with one Programmable Real-Time Unit (PRU) coprocessor (two real-time cores). This PRU can be used for communication protocols such as EtherCAT Master and Slave, PROFINET, Ethernet/IP and Sercos. In this TI Design, EtherCAT Master uses an AM335x Common Platform Ethernet Switch (CPSW) which frees the PRU for other protocols or applications.

図 1 shows the functional block diagram.

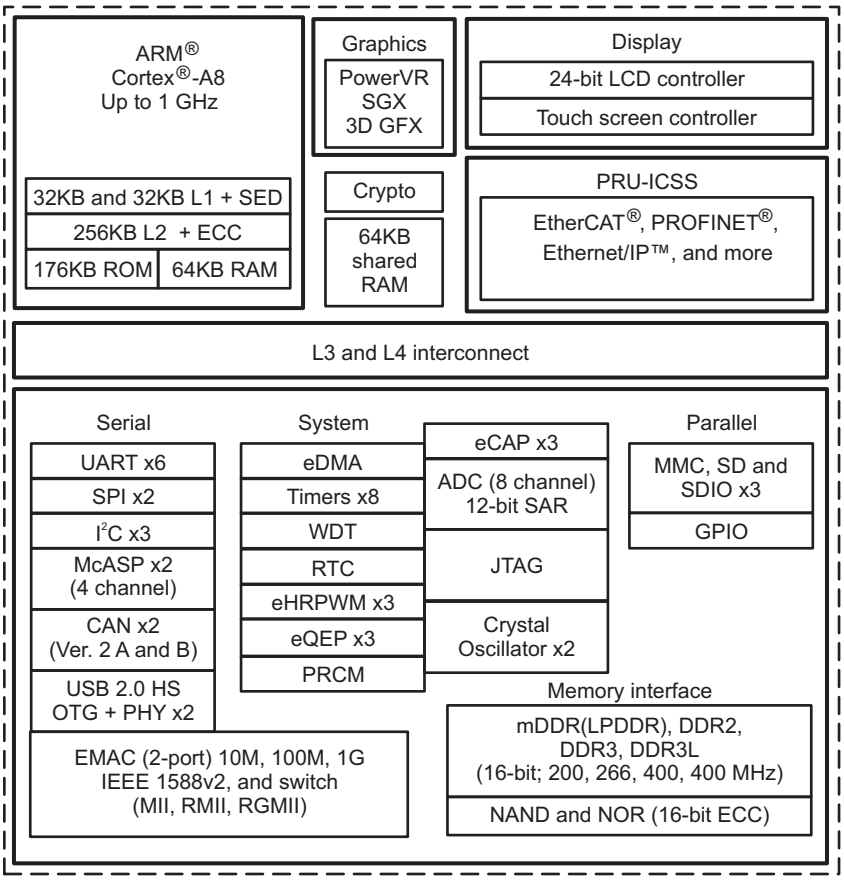


図 1. AM335x Block Diagram

4 acontis EtherCAT® Master Architecture

図 2 shows the module architecture for the acontis EC-Master. The EC-Master stack is divided into the following modules:

- EtherCAT Master Core
  - In the core module, cyclic (process data update) and acyclic (mailbox) EtherCAT commands are sent and received.
- Configuration Layer
  - The EtherCAT Master is configured using an XML file whose format is fixed in the EtherCAT specification ETG.2100. The EC-Master contains an OS-independent XML parser.
- Ethernet Link Layer
  - This layer exchanges Ethernet frames between the master and the slave devices.
- OS Layer
  - All OS-dependent system calls are encapsulated in a small OS layer.

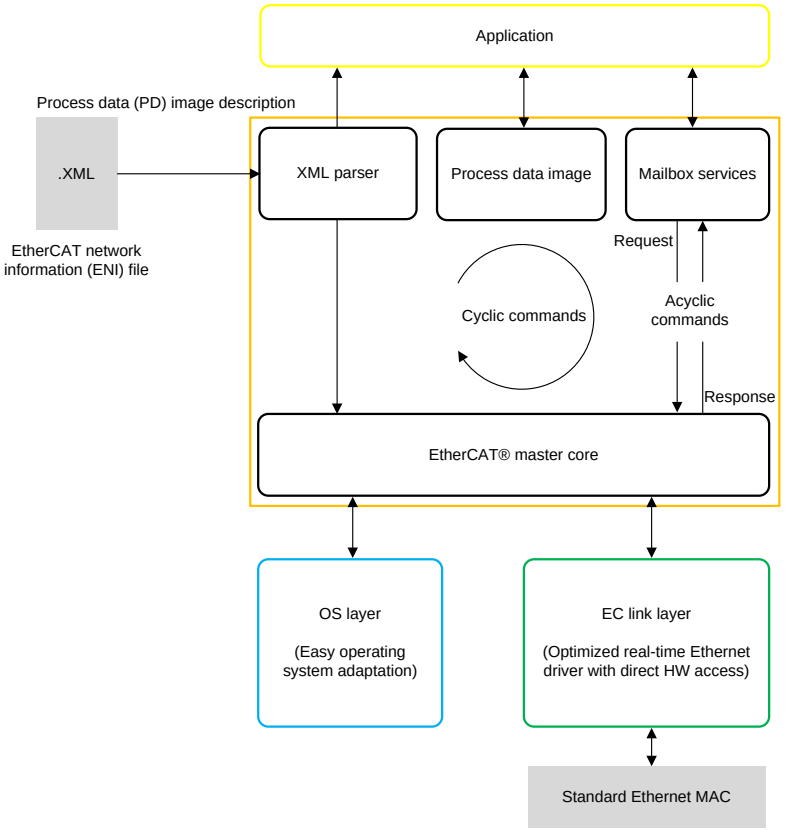


図 2. acontis EC-Master Component Model Architecture

5 Getting Started

5.1 Hardware

The following list shows the hardware required to support this design.

- TMDSICE3359 (AM335x ICE V2 board)
  - Select the CPSW using jumpers J18 and J19 between pin 1 and 2
  - Select pin 1 and 2 shorted on J5 for booting
- EtherCAT Slave device (TMDSIDK437x)

注: This TI Design was tested using the TMDSIDK437x as the EtherCAT slave.

- PC with terminal connection (for example, Tera Term through USB)
- Windows® PC with minimum 2GB RAM

5.2 Software

The following list shows the software required to support this design.

- Acontis EC-Master V2.9 for SYS/BIOS
- AM335x SYS/BIOS Industrial SDK v1.1.0.8

注: After installing Industrial SDK, add or change IA_SDK_HOME in the PC Environment Variables to point to the SDK root directory.

- Code Composer Studio™ 6.1.3 Compiler v5.1 or higher
- SYS/BIOS 6.41.04.54
- XDC v3.30.06.67 or higher
- Serial console terminal application (for example, TeraTerm, minicom, and HyperTerminal)

  If using the AM437x IDK as a slave, download Industrial SDK 2.1.0.1 pre-build binaries

6 Preparing the Application

This TI Design was tested using the AM437x as an EtherCAT Slave. To use the AM437x as the slave device, refer to Appendix A.

The TI Design prebuilt software packages include the ENI file for a topology where the AM437x is the only slave device that is connected. If the user switches to a different slave-bus topology or wants to recreate the ENI files for the AM437x, users can use the acontis EC-Engineer tool. To create an ENI file with the EC-Engineer Tool, refer to Appendix B.

1. Install Industrial SDK version 1.1.0.8 on the host PC
2. Unzip EC_Master_Sysbios_SDK_Eval
3. Consider and select one of the following:
   (a) If the slave bus is the AM437x, copy the MasterENI.c file from the prebuilt package and paste inside <EC-master_installation_path>*Workspace**SYSBIOS*\
   (b) If the slave bus is not the AM437x, convert the ENI file (eni.xml) to a C file with array
      (i) Obtain the eni.xml file by following the instructions in Appendix B
      (ii) Download and install Industrial SDK

(iii) Use the Industrial IDK converter tool found in </ISDK_installation_path>*sdk**tools**bin2header* to convert the eni.xml file

注: The following are example parameters to use with bin2header.exe:

bin2header.exe eni.xml MasterENI.c MasterENI_xml_data

Add the file size information at the end of the new MasterENI.c file. See the following code example.

```
unsigned int MasterENI_xml_data_size = 16426;
```

The file size prints in the console.

4. Open CCS and import the EC-Master demonstration project.
   (a) Navigate to *File→Import→CCS Projects*
   (b) Navigate to <ECmaster_installation_path>*\Acontis_EC_master\EC_Master_Sysbios_SDK_Eval\Workspace\SYSBIOS\EcMasterDemo*
   (c) Click OK, and then click Finish
   (d) Check that SYSBIOS, XDC, and compiler versions are correct in the CCS project properties
   (e) Click the Clean Project option
   (f) Click the Build Project option

The EcMasterDemo.out output application is in the Debug Folder or Release Folder folder after building the project.

注: Users can load and run EcMasterDemo.out from the prebuilt package.

To change the hardcoded parameters for the demonstration, use DEMO_PARAMETERS in ATEMDemoConfig.h. See the following snippet of code.

```
#define DEMO_PARAMETERS    "=auxclk 2000 -v 2 -t 10000 -perf " \
        "-cpsw "
        "1 "    /* port */    \
        "1 "    /* mode */    \
        "1 "    /* priority */    \
        "m "    /* master flag */    \
        "1 " /* PHY address */    \
        "1"    /* PHY connection mode: RGMII */
```

- -auxclk: the clock period in µs
- -t: specifies the time to run the demonstration application in ms
- -perf: enables job-performance measurement

7 Running the Application

Use the following instructions to run the application.

1. Power on the EtherCAT Slave device
2. Confirm that the Ethernet bus is connected to the correct port
   - Find the port that was configured on DEMO_PARAMS; the options are:
     – if Port = 1, use ETH0
     – if Port = 2, use ETH1
   - If using the AM437x as the EtherCAT Slave, connect a cable to PRUETH0 (J6)
3. Connect a USB cable to the ICE v2 board
4. Set up a terminal connection to the Windows PC

5. Configure the host serial port as follows (see 図 3): 115200 baud, no parity, 1 stop bit, no flow control

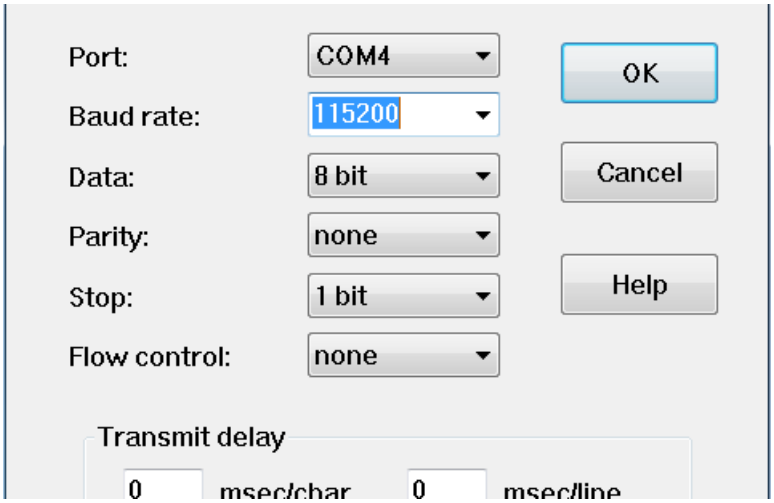


図 3. Serial Port Settings

6. Open CCS
7. Create a target configuration
   (a) Navigate to *View→Target Configuration*.
   (b) Right-click New Target Configuration
   (c) Create a file name (for example, ICEv2.ccxml)
   (d) Set the connection to XDS100v2 USB Emulator
   (e) Set the board or device to ICE_AM3359
   (f) Click Target Configuration and select Cortex-A8
   (g) Enter the initialization script as follows: \..\..\..\am335x_sysbios_ind_sdk_1.1.0.8\sdk\tools\gel\ICE\TMDXICE3359_v2_1A.gel
   (h) Click the Save button
   (i) Right-click ICEv2.ccxml to launch the configuration
   (j) Right-click Cortex-A8 to connect to the target
   (k) Click CPU (Reset HW)
   (l) Click on Scripts
   (m) Click AM335x System Initialization
   (n) Click AM3359_ICE_Initialization
   (o) Click on Load Program, and then click Browse Project
   (p) Click EcMasterDemo
   (q) Click on Debug or Release
   (r) Click EcMasterDemo.out
   (s) Click the Resume button

See Appendix C for an example of an output console.

8 EtherCAT® Master Benchmark

In this design, the EtherCAT Master has no internal tasks. As a result, EtherCAT functions as a driver for the application. This implementation brings some benefits such as no synchronization issues between application and EtherCAT Master. The application controls the timing of events, enabling the cyclic portion to run within an interrupt service routine (ISR).

To benchmark the EtherCAT Master on AM335x, the test includes seven commercially-available slave devices (EK110, 2x EL2004, 2x EL1004, EL4132, EK1110). The configuration used a frame load size of 579 bytes with 512 bytes of process data and a mailbox transfer to EL4132. The EtherCAT Master ran for 10 s, and TSC profiling data was collected for the following cycle jobs (see 図 4).

- I: EtherCAT Master refresh inputs
- O: EtherCAT Master writer outputs
- A: EtherCAT Master administration functions
- M: EtherCAT Master acyclic datagrams and commands
- App: The application processes inputs and creates output values

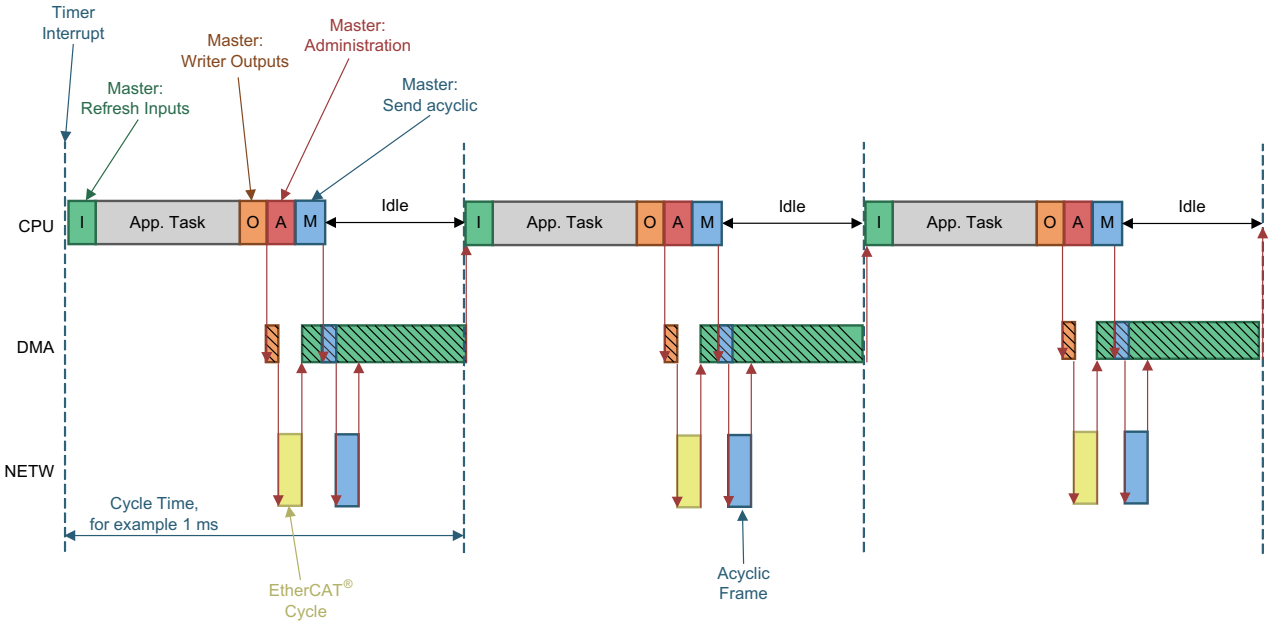


図 4. Bus Timing Diagram

9 Test Data

表 1 shows the measurements (average CPU load) in 図 4.

表 1. AM335x EC-Master CPU performance

Sitara AM335 600 MHz NIC: CPSW TI-RTOS (SYS/BIOS)		
NUMBER	**EC-MASTER JOB**	**AVERAGE µs**
1	I: Process Inputs	16
2	O: Send Outputs	10

表 1. AM335x EC-Master CPU performance (continued)

Sitara AM335 600 MHz NIC: CPSW TI-RTOS (SYS/BIOS)		
NUMBER	**EC-MASTER JOB**	**AVERAGE µs**
3	A: Administration	9
4	M: Send Acyclic Frame	4
	Total CPU Time	39

10 Design Files

10.1 Bill of Materials

To download the bill of materials (BOM), see the design files at TIDEP0043.

10.2 Software Files

To download the software files, see the design files at the TIDEP0043.

11 References

1. EtherCAT.org, *Technical Introduction and Overview*, (EtherCAT)
2. Texas Instruments, *EtherCAT ® on Sitara ™ Processors*, White Paper (SPRY187)

11.1 商標

12 About the Author

PAULA CARRILLO is a software engineer for the Embedded Processing group at TI. She obtained her MSEE from Florida Atlantic University and her Bachelor's Degree at Javeriana University, Colombia. Since joining TI in 2009, Paula has been working on different SoC and multicore DSP platforms developing applications for high-performance video codecs, synthetic aperture radar (SAR), and industrial communication protocols.

13 About the Author—acontis technologies

STEFAN ZINTGRAF graduated as a Software Engineer in 1987 and began working as software engineer in Sulzer and LP Elektronik. After several years working as team leader in LP Elektronik he co-founded acontis technologies GmbH in 2001, working as general manager. Stefan took over acontis' EtherCAT development activities in 2005 after being responsible for the acontis Windows® real-time products. He is also responsible for the fast growing EtherCAT market in Asia in worldwide sales activities.

改訂履歴

資料番号末尾の英字は改訂を表しています。その改訂履歴は英語版に準じています。

2015年9月発行のものから更新 **Page**

Appendix A AM437x Running EtherCAT Slave

A.1 Preparing the SD Card

Prepare the SD card by file allocation table (FAT) formatting it as follows:

1. Ensure the HP USB disk storage format tool v2.0.6 is portable
2. Run the HP USB disk storage format tool v2.0.6 as portable executable. The executable detects the SD card that is plugged into the reader. If undetected, direct the executable to the new disk.
3. Choose FAT32 if the SD card is greater than 4GB. If not, use FAT
4. Click the Start button

注: After formatting, the card can be populated by the files.

5. Install the ISDK prebuilt binaries (Industrial SDK 2.1.0.1)
6. Copy MLO from <Installation_path>*sysbios_ind_sdk_prebuilt_02_01_00_01**Bootloader**SD**am437x_release* to the SD card
7. Copy the EtherCAT Slave application from <Installation_path>*sysbios_ind_sdk_prebuilt_02_01_00_01**ethercat_slave**am437x_release*
8. Connect the Ethernet cable to the ICCS port J6
9. Use Wireshark to test the ecat (EtherCAT) packets

   Alternatively, use an EtherCAT Master, such as TwinCAT®, to test master-to-slave connectivity

For details using TwinCAT with TI IDK boards, see the user guide at wiki.ti.com.

Appendix B acontis EC-Engineer Tool for Creating an .ENI File

1. Register for a free evaluation version at www.acontis.com
2. Install the EC-Engineer tool on the PC
3. Connect the EtherCAT slave to the computer (if using the AM437x, connect the Ethernet cable to J6)
4. Open the EC-Engineer tool
5. Select Online Configuration
6. Select the EtherCAT Master unit (Class A) as the master unit
7. Click the OK button
8. Select 2000 as the Cycle Time (µs)
9. Select the desired network adapter as the slave that is connected to the local system.

注: If using the AM437x as the slave, add TI's IDK.xml to the ESI manager as follows:

- Open the ESI manager
- Add the IDK.xml file
- Browse to <Installation_path>*sysbios_ind_sdk_2.1.0.1**sdk**examples**ethercat_slave**esi**TiEtherCATLib.xml*
- Open the file
- Navigate to the Network option
- Click Scan EtherCAT Network
- Click Export ENI after the slaves are found

Selecting Export ENI exports eni.xml.

Appendix C Application Console Output

```
TI Industrial SDK Version - IASDK 1.1.0.8
Device name : AM3359
Chip Revision : AM335x ES1.2 [PG2.1]

SYS/BIOS EcMaster Sample application running on ICE V2
Full command line: -auxclk 2000 -v 2 -t 100000 -perf -cpsw 1 1 1 m 1 1

Run demo now with cycle time 2000 usec Using AuxClock
==========================
Initialize EtherCAT Master
==========================
EC-Master V2.7.2.12 (Eval) for SYSBIOS Copyright acontis technologies GmbH @ 2015
CPSW INF: Port 1, Prio 1, Flags [Polling] [Master], MAC 00:00:00:00:00:01

CPSW INF: CPSW3G found. CPSW INF: HW-Id: 0x0019, RTL: 0, Major: 1, Minor: 0xc
CPSW INF: PHY found. Id=0x2000a211
CPSW INF: Restart PHY auto negotiation
CPSW INF: PHY auto negotiation completed
Evaluation Version, stop sending ethernet frames after 60 minutes!
PDI Watchdog expired - Slave Slave_1001 [TIESC-002]: - EtherCAT address=1001
Bus scan successful - 1 slaves found

********************************************************************************
Number : 0
Vendor ID : 0xE000059D = ----
Product Code: 0x54490002 = Unknown
Revision : 0x00000001 Serial Number: 0
ESC Type : TI Sitara (0x90) Revision: 2 Build: 947
Bus AutoInc Address: 0 (0x0)
Bus Station Address: 1001 (0x3e9)
Bus Alias Address : 0000 (0x0)
Connection at Port 0: yes Port 1: no Port 2: no Port 3: no
SlaveID at Port 0: 65536 Port 1: -1 Port 2: -1 Port 3: -1
Config Station Address: 1001 (0x3e9)
```

```
PD IN Byte.Bit offset: 0.0 Size: 32 bits
PD OUT Byte.Bit offset: 0.0 Size: 32 bits
EtherCAT network adapter MAC: 00-00-00-00-00-01

====================
Start EtherCAT Master
====================
Master state changed from <UNKNOWN> to <INIT>
Master state changed from <INIT> to <PREOP>
Master state changed from <PREOP> to <SAFEOP>
Master state changed from <SAFEOP> to <OP>

Job times during startup <INIT> to <OP>:
================================================================
PerfMsmt 'JOB_ProcessAllRxFrames' (avg/max) [usec]: 29.6/ 91.4
PerfMsmt 'JOB_SendAllCycFrames ' (avg/max) [usec]: 18.2/ 36.5
PerfMsmt 'JOB_MasterTimer ' (avg/max) [usec]: 50.7/263.0
PerfMsmt 'JOB_SendAcycFrames ' (avg/max) [usec]: 21.3/ 64.4
PerfMsmt 'Cycle Time ' (avg/max) [usec]: 1624.0/2011.7
PerfMsmt 'myAppWorkPd ' (avg/max) [usec]: 1.7/ 5.8

================================================================
PerfMsmt 'JOB_ProcessAllRxFrames' (avg/max) [usec]: 32.9/ 48.4
PerfMsmt 'JOB_SendAllCycFrames ' (avg/max) [usec]: 30.6/ 39.2
PerfMsmt 'JOB_MasterTimer ' (avg/max) [usec]: 52.3/ 61.5
PerfMsmt 'JOB_SendAcycFrames ' (avg/max) [usec]: 7.0/ 15.9
PerfMsmt 'Cycle Time ' (avg/max) [usec]: 1998.1/2014.2
PerfMsmt 'myAppWorkPd ' (avg/max) [usec]: 2.2/ 9.0
```

Appendix D Troubleshooting

If the code hangs while reading PHY register MDIO, add the following line of code in main.c:

```
GPIOInit();
UTILsSetBoardType(3); //Added to fix board type to ICEv2
s_boardType = UTILsGetBoardType();
s_uartInstance = InitUart(s_boardType)
```

If the following error is printed in the console, the SPI likely preflashed.

```
001181 : CPSW ERR: mdio ACK missing
001181 : CPSW INF: mdio ref 2 error
001181 : CPSW ERR: PHY initialization failed
```

The two solutions to this error are:

- Change the boot jumper to NOR Pin 1 and 2 shorted on J5, or
- Erase SPI flash
  - For assistance erasing flash memory, see wiki.ti.com

For additional troubleshooting, refer to section 4.2 Error Codes in *Acontis EC-Master Stack Class B Version 2.7 Document* by navigating to <ECmaster_installation_path>*\Acontis_EC_master\EC_Master_Sysbios_SDK_Eval\Doc*.

Users should also refer to *AM335 Sysbios User Guide and Getting Started Guide* by navigating to </ASDK_path>*\am335x_sysbios_ind_sdk_1.1.0.8\sdk\docs*.

TIの設計情報およびリソースに関する重要な注意事項

Texas Instruments Incorporated ("TI")の技術、アプリケーションその他設計に関する助言、サービスまたは情報は、TI製品を組み込んだアプリケーションを開発する設計者に役立つことを目的として提供するものです。これにはリファレンス設計や、評価モジュールに関係する資料が含まれますが、これらに限られません。以下、これらを総称して「TIリソース」と呼びます。いかなる方法であっても、TIリソースのいずれかをダウンロード、アクセス、または使用した場合、お客様(個人、または会社を代表している場合にはお客様の会社)は、これらのリソースをここに記載された目的にのみ使用し、この注意事項の条項に従うことに合意したものとします。

TIによるTIリソースの提供は、TI製品に対する該当の発行済み保証事項または免責事項を拡張またはいかなる形でも変更するものではなく、これらのTIリソースを提供することによって、TIにはいかなる追加義務も責任も発生しないものとします。TIは、自社のTIリソースに訂正、拡張、改良、およびその他の変更を加える権利を留保します。

お客様は、自らのアプリケーションの設計において、ご自身が独自に分析、評価、判断を行う責任がお客様にあり、お客様のアプリケーション(および、お客様のアプリケーションに使用されるすべてのTI製品)の安全性、および該当するすべての規制、法、その他適用される要件への遵守を保証するすべての責任をお客様のみが負うことを理解し、合意するものとします。お客様は、自身のアプリケーションに関して、(1) 故障による危険な結果を予測し、(2) 障害とその結果を監視し、および、(3) 損害を引き起こす障害の可能性を減らし、適切な対策を行う目的での、安全策を開発し実装するために必要な、すべての技術を保持していることを表明するものとします。お客様は、TI製品を含むアプリケーションを使用または配布する前に、それらのアプリケーション、およびアプリケーションに使用されているTI製品の機能性を完全にテストすることに合意するものとします。TIは、特定のTIリソース用に発行されたドキュメントで明示的に記載されているもの以外のテストを実行していません。

お客様は、個別のTIリソースにつき、当該TIリソースに記載されているTI製品を含むアプリケーションの開発に関連する目的でのみ、使用、コピー、変更することが許可されています。明示的または黙示的を問わず、禁反言の法理その他どのような理由でも、他のTIの知的所有権に対するその他のライセンスは付与されません。また、TIまたは他のいかなる第三者のテクノロジまたは知的所有権についても、いかなるライセンスも付与されるものではありません。付与されないものには、TI製品またはサービスが使用される組み合わせ、機械、プロセスに関連する特許権、著作権、回路配置利用権、その他の知的所有権が含まれますが、これらに限られません。第三者の製品やサービスに関する、またはそれらを参照する情報は、そのような製品またはサービスを利用するライセンスを構成するものではなく、それらに対する保証または推奨を意味するものでもありません。TIリソースを使用するため、第三者の特許または他の知的所有権に基づく第三者からのライセンス、あるいはTIの特許または他の知的所有権に基づくTIからのライセンスが必要な場合があります。

TIのリソースは、それに含まれるあらゆる欠陥も含めて、「現状のまま」提供されます。TIは、TIリソースまたはその仕様に関して、明示的か暗黙的かにかかわらず、他のいかなる保証または表明も行いません。これには、正確性または完全性、権原、続発性の障害に関する保証、および商品性、特定目的への適合性、第三者の知的所有権の非侵害に対する黙示の保証が含まれますが、これらに限られません。

TIは、いかなる苦情に対しても、お客様への弁護または補償を行う義務はなく、行わないものとします。これには、任意の製品の組み合わせに関連する、またはそれらに基づく侵害の請求も含まれますが、これらに限られず、またその事実についてTIリソースまたは他の場所に記載されているか否かを問わないものとします。いかなる場合も、TIリソースまたはその使用に関連して、またはそれらにより発生した、実際的、直接的、特別、付随的、間接的、懲罰的、偶発的、または、結果的な損害について、そのような損害の可能性についてTIが知らされていたかどうかにかかわらず、TIは責任を負わないものとします。

お客様は、この注意事項の条件および条項に従わなかったために発生した、いかなる損害、コスト、損失、責任からも、TIおよびその代表者を完全に免責するものとします。

この注意事項はTIリソースに適用されます。特定の種類の資料、TI製品、およびサービスの使用および購入については、追加条項が適用されます。これには、半導体製品(http://www.ti.com/sc/docs/stdterms.htm)、評価モジュール、およびサンプル(http://www.ti.com/sc/docs/sampterms.htm)についてのTIの標準条項が含まれますが、これらに限られません。