How to Use Green Box Platform for Powertrain and VDS Applications

Scott O'Brien

Field Applications Engineer

With material provided from Shanaka Yapa Appuhamillage Don and Sam Moser

October 2018 | AMF-AUT-T3383

SECURE CONNECTIONS FOR A SMARTER WORLD

Agenda

- GreenBox Overview
- Hardware
- Engine Control Demo
- Inverter Control Demo
- Software
- Documentation and Reference Material

GreenBox S32SDEVPL-KITP

GreenBox Emulation System Overview

Pre-silicon SW development for advanced vehicle dynamics and electrification

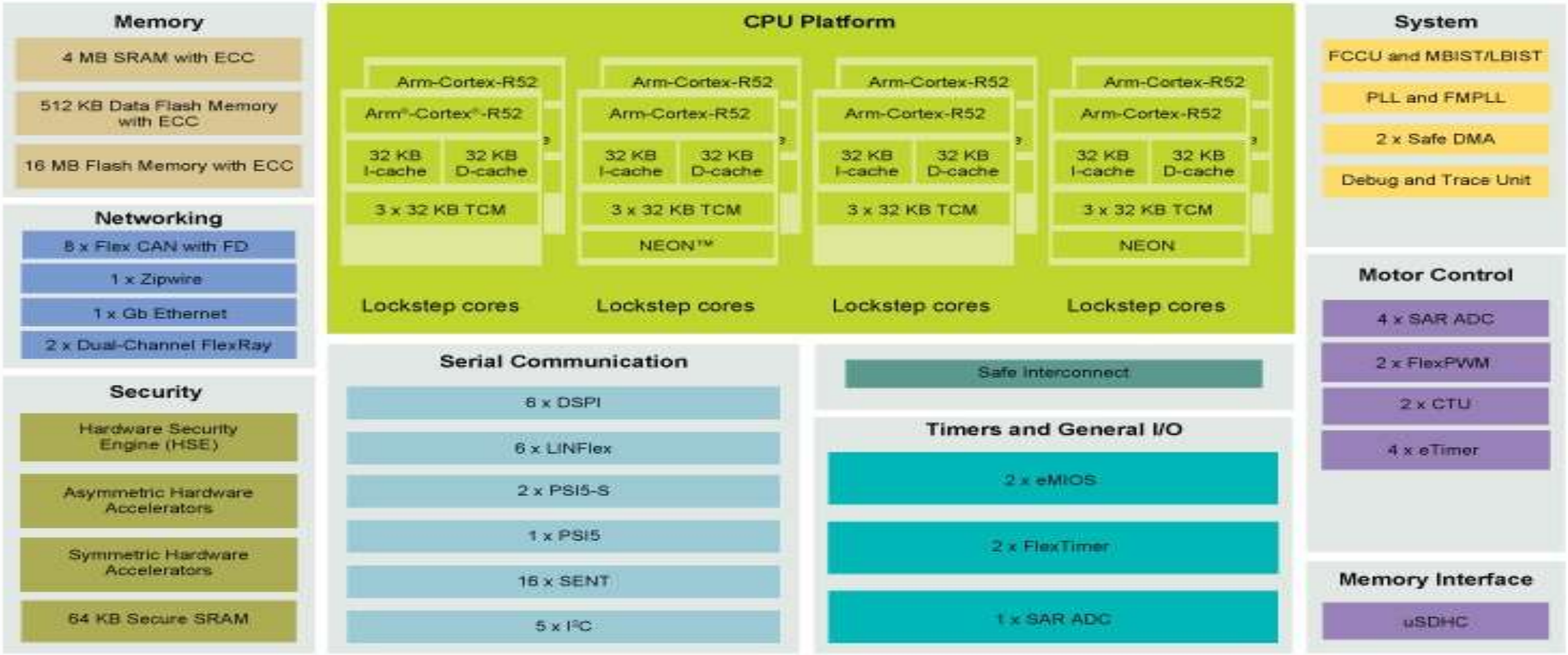
Low Cost Emulation of NXP Next Gen Devices
Using existing NXP solutions

Early ARM Core Development
Cortex A53 sw development & tools
NEON SIMD extension
Multi-core development with sw partitioning
Established ARM toolchain

Standard Application Support Software
Engine Management
Motor Control

GreenBox Kit	Compute Core	Peripherals	Applications
S32PDEVPL-KITC	S32V234 (A53)	MPC5777C	Next Gen Powertrain HEV/EV Electrification
S32SDEVPL-KITP	S32V234 (A53)	MPC5744P	Domain Control - ADAS, Powertrain, Safety Braking, Chassis Motor Control Safety

Next Gen S32S Processor Block Diagram



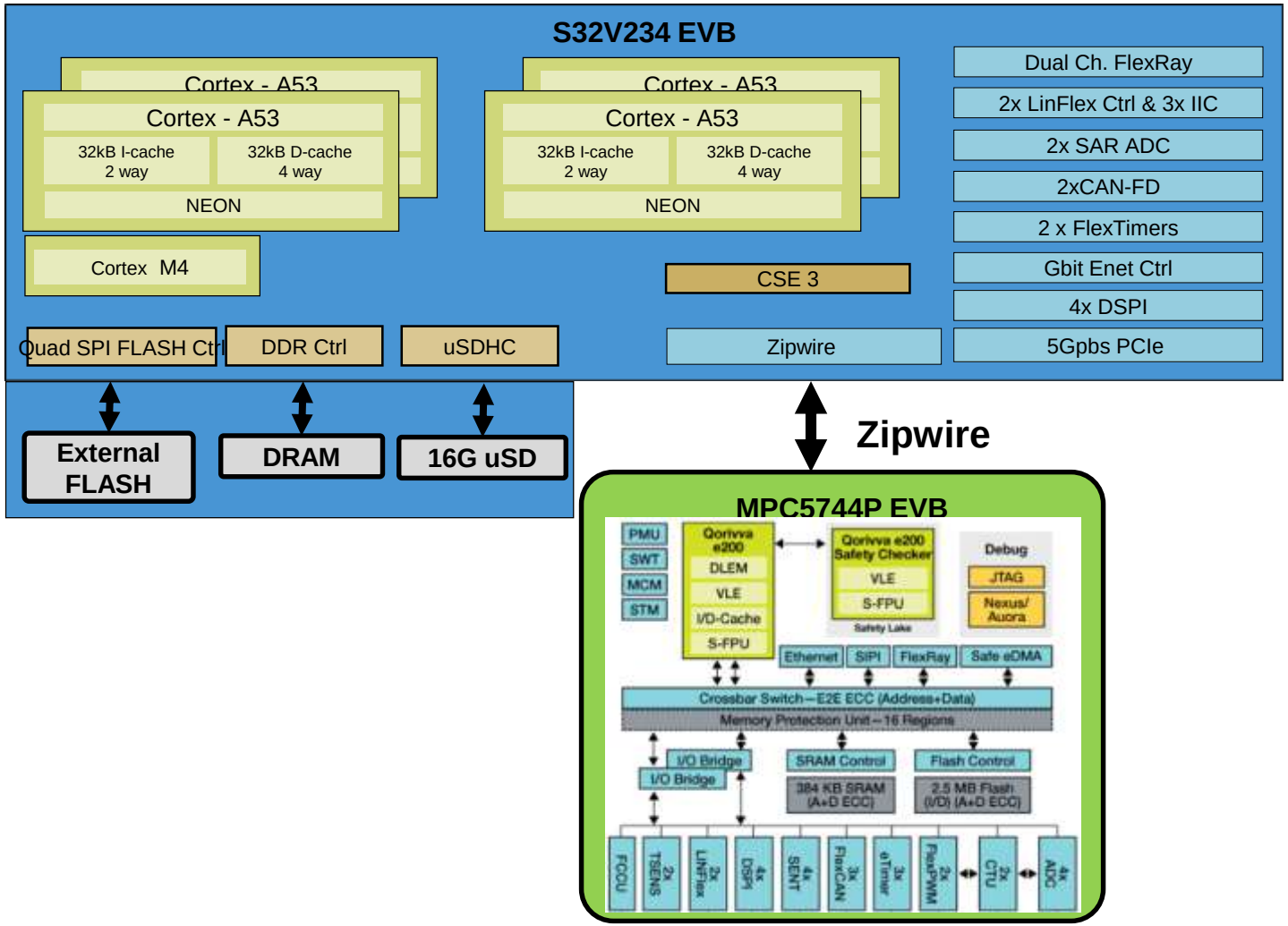
S32S/P Hardware Emulator Overview

S32V234 Main Board

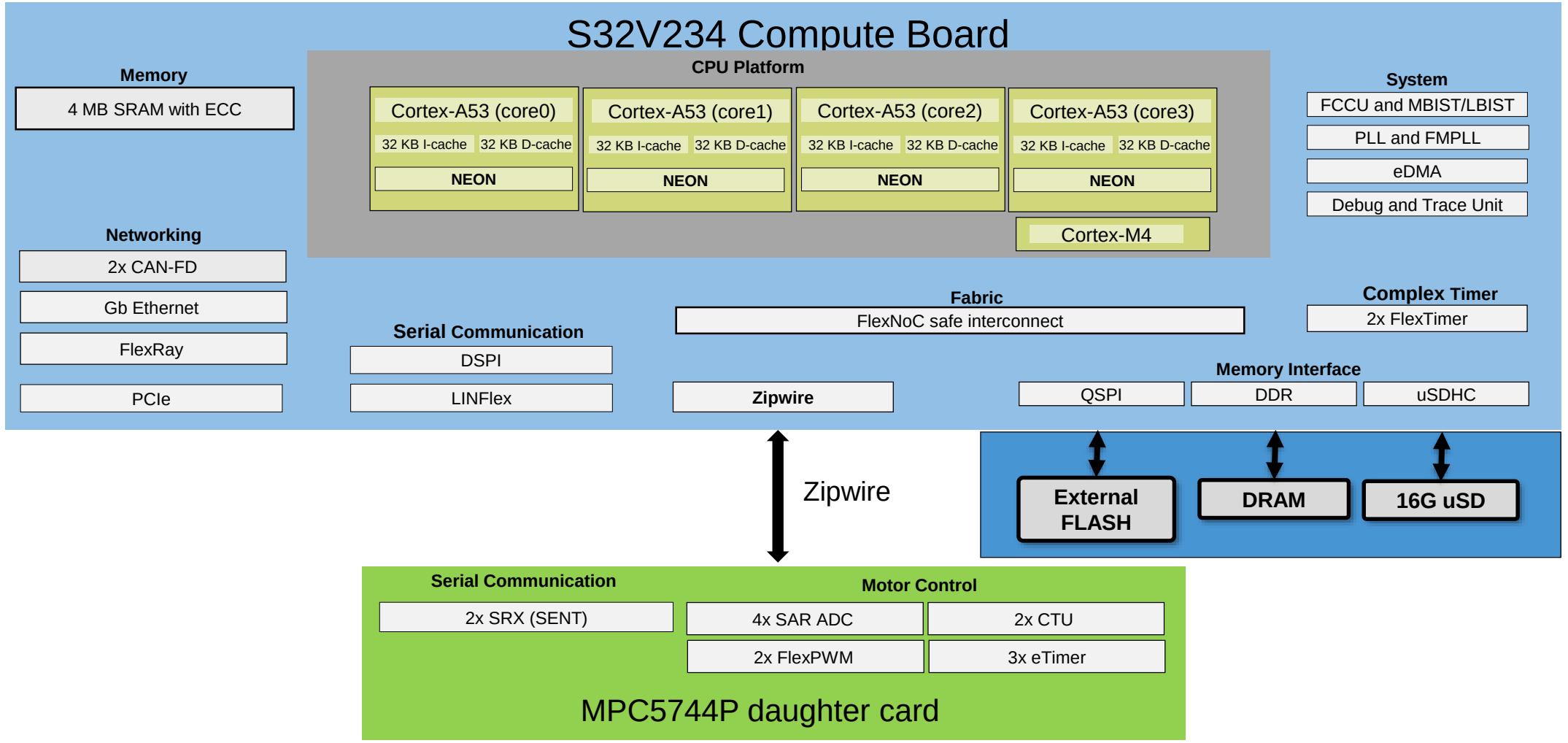
- Emulates 16nm core compute complex + peripherals
- A53 cores w/NEON
- 1x M4 core

MPC5744P or MPC5777C Daughter Card

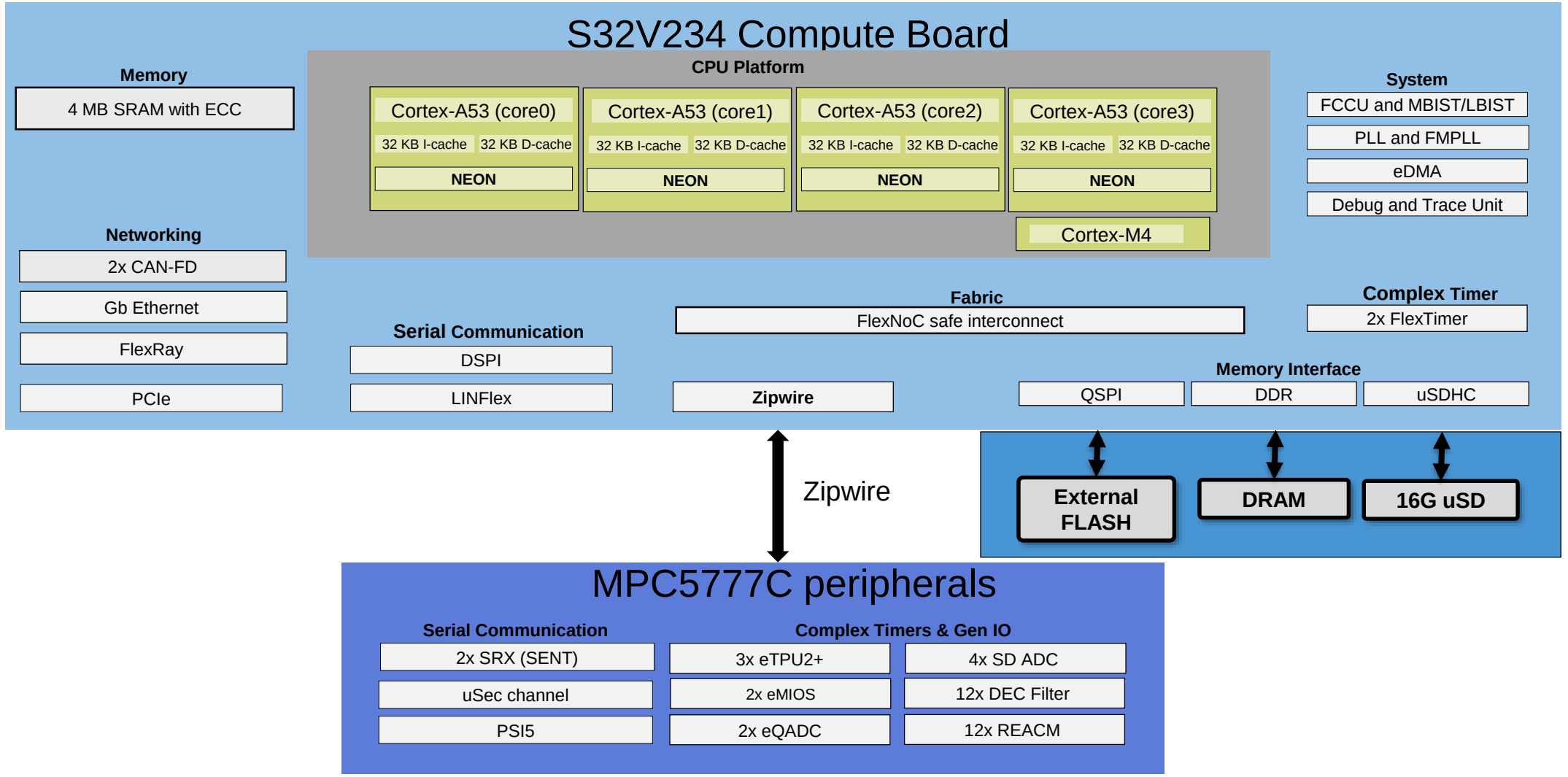
- Emulates application specific peripherals
- Motor Control, Safety
- Engine management



S32S247 Emulation – S32V234 + MPC5744P



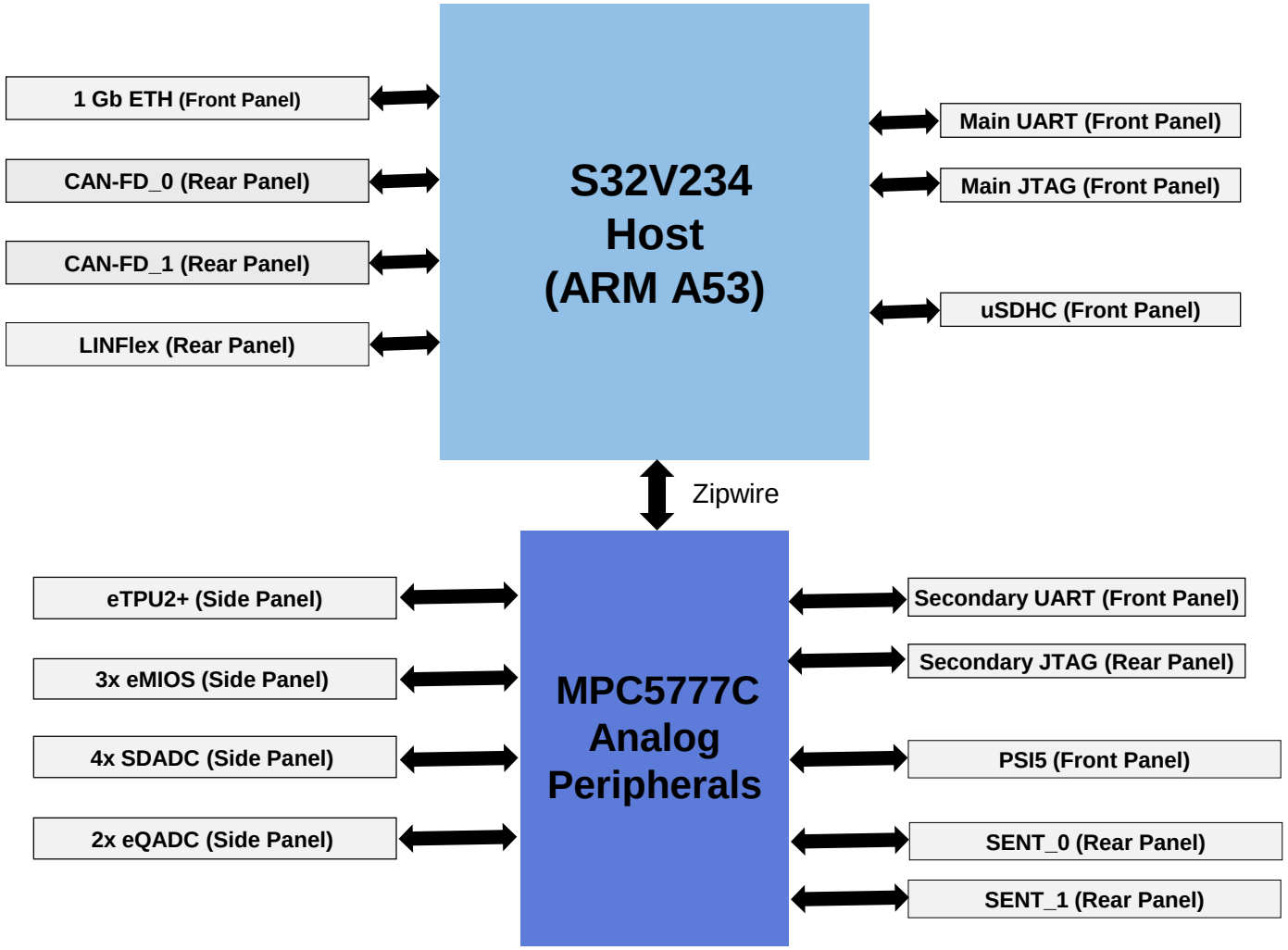
S32P247 Emulation – S32V234 + MPC5777C



Hardware

GreenBox S32SDEVPL-KITP

S32SPDEVPL-KIT Hardware Overview



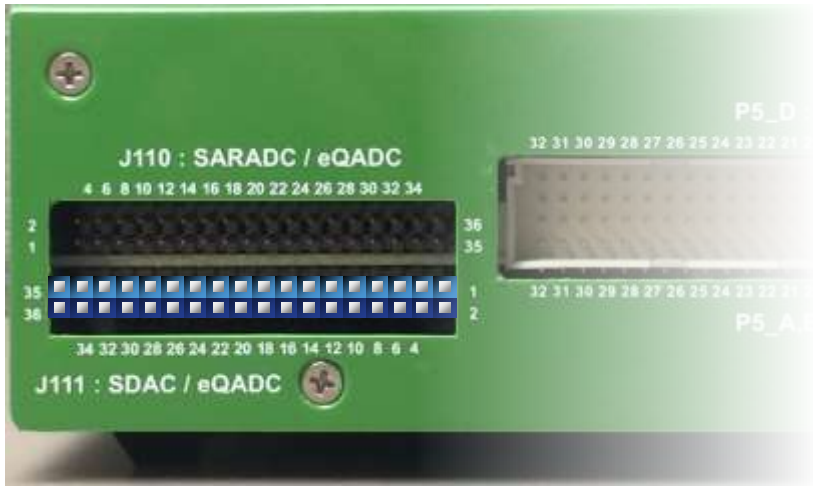
S32PDEVPL-KITC: Pinout

J110

J110

FUNCTION	PIN		PIN	FUNCTION
ANA16_SDB0	**J110-01**		**J110-02**	ANB8
ANA17_SDB1	**J110-03**		**J110-04**	ANB9
ANA18_SDB2	**J110-05**		**J110-06**	ANB10
ANA19_SDB3	**J110-07**		**J110-08**	ANB11
ANA20_SDC0	**J110-09**		**J110-10**	ANB12
ANA21_SDC1	**J110-11**		**J110-12**	ANB13
ANA22_SDC2	**J110-13**		**J110-14**	ANB14
ANA23_SDC3	**J110-15**		**J110-16**	ANB15
ANB0_SDD0	**J110-17**		**J110-18**	ANB16
ANB1_SDD1	**J110-19**		**J110-20**	ANB17
ANB2_SDD2	**J110-21**		**J110-22**	ANB18
ANB3_SDD3	**J110-23**		**J110-24**	ANB19
ANB4_SDD4	**J110-25**		**J110-26**	ANB20
ANB5_SDD5	**J110-27**		**J110-28**	ANB21
ANB6_SDD6	**J110-29**		**J110-30**	ANB22
ANB7_SDD7	**J110-31**		**J110-32**	ANB23
	J110-33		**J110-34**	
	J110-35		**J110-36**	

S32PDEVPL-KITC: Pinout

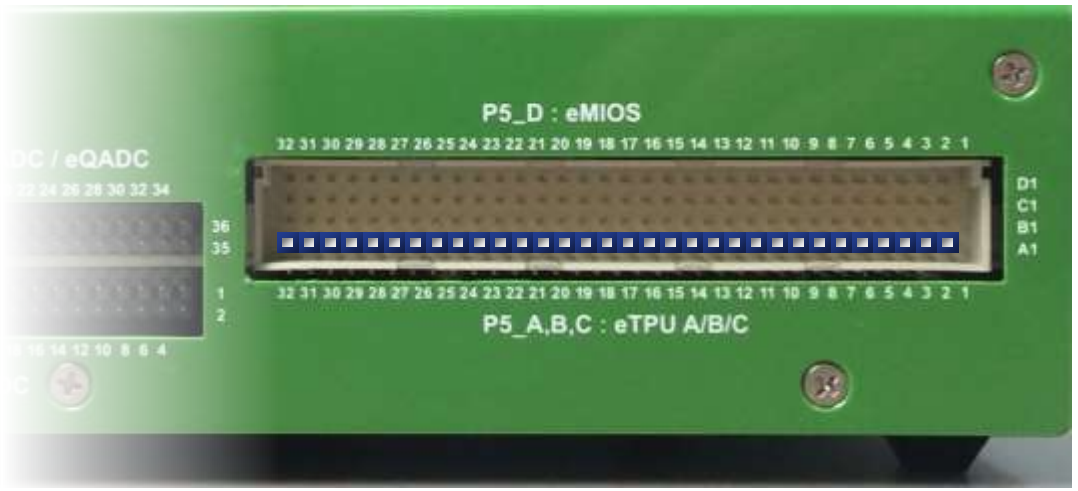


J111

J111

FUNCTION	PIN	PIN	FUNCTION
AN24	**J110-01**	**J110-02**	ANA0_SDA0
AN25	**J110-03**	**J110-04**	ANA1_SDA1
AN26	**J110-05**	**J110-06**	ANA2_SDA2
AN27	**J110-07**	**J110-08**	ANA3_SDA3
AN28	**J110-09**	**J110-10**	ANA4
AN29	**J110-11**	**J110-12**	ANA5
AN30	**J110-13**	**J110-14**	ANA6
AN31	**J110-15**	**J110-16**	ANA7
AN32	**J110-17**	**J110-18**	ANA8
AN33	**J110-19**	**J110-20**	ANA9
AN34	**J110-21**	**J110-22**	ANA10
AN35	**J110-23**	**J110-24**	ANA11
AN36	**J110-25**	**J110-26**	ANA12
AN37	**J110-27**	**J110-28**	ANA13
AN38	**J110-29**	**J110-30**	ANA14
AN39	**J110-31**	**J110-32**	ANA15
	J110-33	**J110-34**	
	J110-35	**J110-36**	

S32PDEVPL-KITC: Pinout

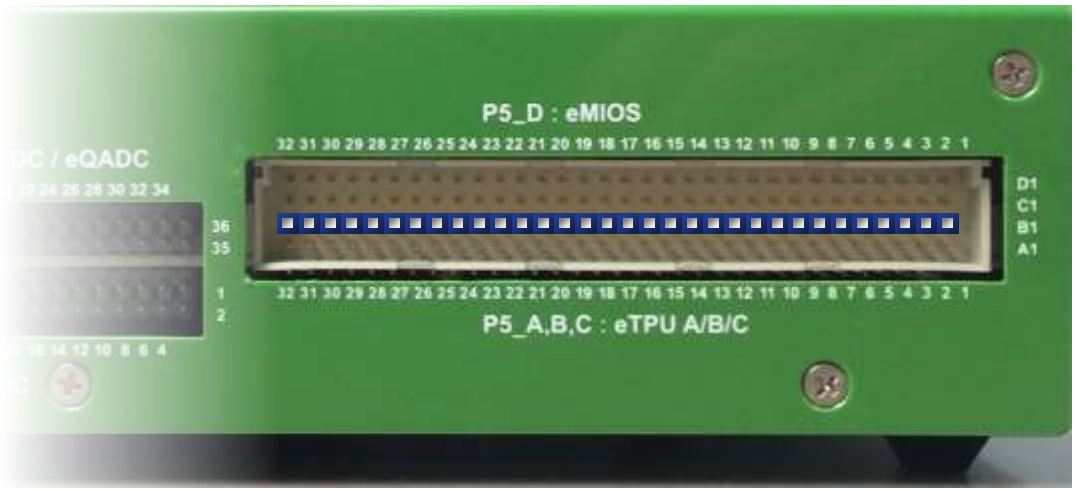


P5A

P5A

PIN	FUNCTION	PCR#
P5A-01	ETPUA0	114
P5A-02	ETPUA1	115
P5A-03	ETPUA2	116
P5A-04	TP55	
P5A-05	ETPUA4	118
P5A-06	ETPUA5	119
P5A-07	ETPUA6	120
P5A-08	ETPUA7	121
P5A-09	ETPUA8	122
P5A-10	ETPUA9	123
P5A-11	ETPUA10	124
P5A-12	TCRCLKA	113
P5A-13	ETPUA12	126
P5A-14	ETPUA13	127
P5A-15	ETPUA14	128
P5A-16	ETPUA15	129
P5A-17	ETPUA16	130
P5A-18	ETPUA17	131
P5A-19	ETPUA18	132
P5A-20	ETPUA19	133
P5A-21	ETPUA20	134
P5A-22	ETPUA21	135
P5A-23	TP51	
P5A-24	GND	
P5A-25	GND	
P5A-26	GND	
P5A-27	GND	
P5A-28	ETPUA27	141
P5A-29	ETPUA28	142
P5A-30	ETPUA29	143
P5A-31	ETPUA30	144
P5A-32	ETPUA31	145

S32PDEVPL-KITC: Pinout

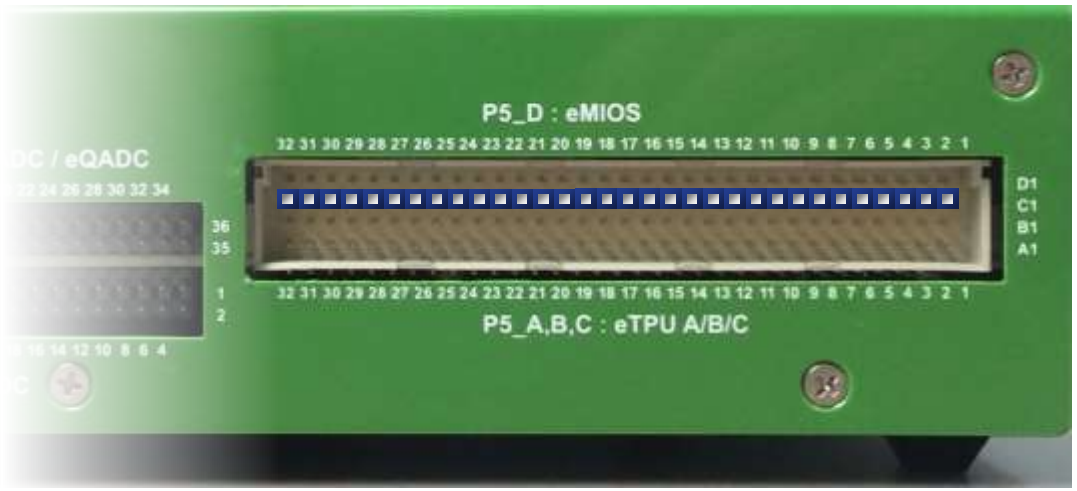


P5B

P5B

PIN	FUNCTION	PCR#
P5B-01	ETPUB0	147
P5B-02	ETPUB1	148
P5B-03	ETPUB2	149
P5B-04	ETPUB3	150
P5B-05	ETPUB4	151
P5B-06	ETPUB5	152
P5B-07	ETPUB6	153
P5B-08	ETPUB7	154
P5B-09	ETPUB8	155
P5B-10	ETPUB9	156
P5B-11	ETPUB10	157
P5B-12	ETPUB11	158
P5B-13	ETPUB12	159
P5B-14	ETPUB13	160
P5B-15	ETPUB14	161
P5B-16	ETPUB15	162
P5B-17	ETPUB16	163
P5B-18	ETPUB17	164
P5B-19	ETPUB18	165
P5B-20	ETPUB19	166
P5B-21	ETPUB20	167
P5B-22	ETPUB21	168
P5B-23	ETPUB22	169
P5B-24	ETPUB23	170
P5B-25	ETPUB24	171
P5B-26	ETPUB25	172
P5B-27	ETPUB26	173
P5B-28	ETPUB27	174
P5B-29	ETPUB28	175
P5B-30	ETPUB29	176
P5B-31	ETPUB30	177
P5B-32	ETPUB31	178

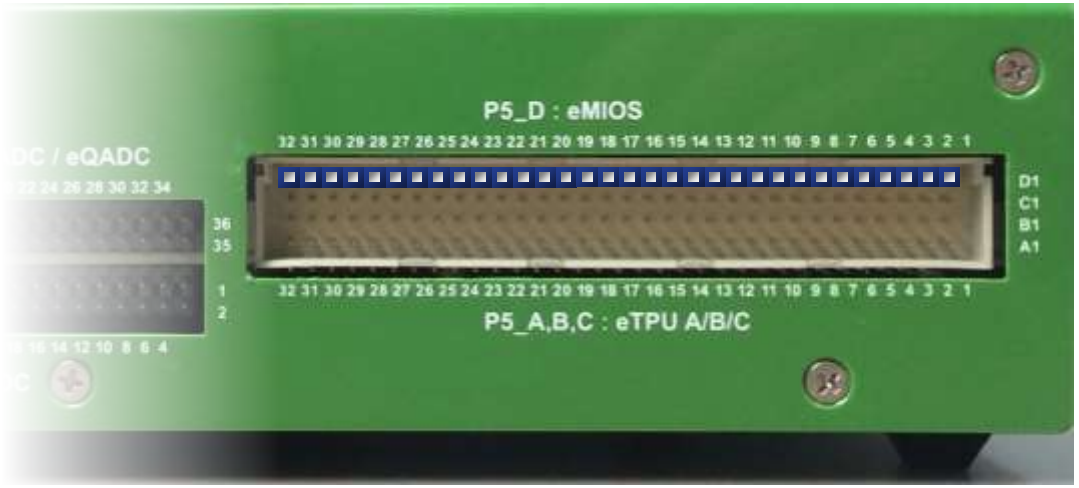
S32PDEVPL-KITC: Pinout



P5C

PIN	FUNCTION	PCR#
P5C-01	TP52	
P5C-02	TP56	
P5C-03	ETPUC2	443
P5C-04	ETPUC3	444
P5C-05	ETPUC4	445
P5C-06	TP57	
P5C-07	TP58	
P5C-08	ETPUC7	448
P5C-09	ETPUC8	449
P5C-10	ETPUC9	450
P5C-11	ETPUC10	451
P5C-12	TCRCLKC	440
P5C-13	TCRCLKB	146
P5C-14	GND	
P5C-15	GND	
P5C-16	ETPUC15	456
P5C-17	ETPUC16	457
P5C-18	ETPUC17	458
P5C-19	ETPUC18	459
P5C-20	GND	
P5C-21	GND	
P5C-22	ETPUC21	462
P5C-23	ETPUC22	463
P5C-24	ETPUC23	464
P5C-25	ETPUC24	465
P5C-26	ETPUC25	466
P5C-27	ETPUC26	467
P5C-28	ETPUC27	468
P5C-29	ETPUC28	469
P5C-30	ETPUC29	470
P5C-31	ETPUC30	471
P5C-32	ETPUC31	472

S32PDEVPL-KITC: Pinout



P5D

P5D

	PIN	FUNCTION	PCR#
	P5D-01	EMIOS0	179
	P5D-02	EMIOS1	180
	P5D-03	EMIOS2	181
	P5D-04	EMIOS3	182
	P5D-05	EMIOS4	183
	P5D-06	EMIOS5	184
	P5D-07	EMIOS6	185
	P5D-08	EMIOS7	186
	P5D-09	EMIOS8	187
	P5D-10	EMIOS9	188
	P5D-11	EMIOS10	189
	P5D-12	EMIOS11	190
	P5D-13	EMIOS12	191
	P5D-14	EMIOS13	192
	P5D-15	EMIOS14	193
	P5D-16	EMIOS15	194
	P5D-17	EMIOS16	195
	P5D-18	EMIOS17	196
	P5D-19	EMIOS18	197
	P5D-20	EMIOS19	198
	P5D-21	EMIOS20	199
	P5D-22	EMIOS21	200
	P5D-23	EMIOS22	201
	P5D-24	EMIOS23	202
	P5D-25	EMIOS24	203
	P5D-26	EMIOS25	204
	P5D-27	EMIOS26	432
	P5D-28	EMIOS27	433
	P5D-29	EMIOS28	434
	P5D-30	EMIOS29	435
	P5D-31	EMIOS30	436
	P5D-32	EMIOS31	437

S32PDEVPL-KITC: Communication Interfaces

PSI5

FUNCTION	PIN
PSI5A_DOUT	**PSI5-01**
PSI5A_DIN	**PSI5-02**
PSI5B_DOUT	**PSI5-03**
PSI5B_DIN	**PSI5-04**

SENT_0

FUNCTION	PIN
3.3V	**SENT0-01**
SENT0_A	**SENT0-02**
SENT0_B	**SENT0-03**
GND	**SENT0-04**

SENT_1

FUNCTION	PIN
3.3V	**SENT1-01**
SENT1_A	**SENT1-02**
SENT1_B	**SENT1-03**
GND	**SENT1-04**

LIN

FUNCTION	PIN	PIN	FUNCTION
LIN	**PSI5-04**	**PSI5-03**	LIN_VSUP_1
GND	**PSI5-02**	**PSI5-01**	GND

S32PDEVPL-KITC: Power and Reset

Power supply specifications:
S32PDEVPL-KITC is powered through an external 12V power supply. The power supply is included with the kit.

Requirements:
Fully regulated Switching Power Supply
Input Voltage 100-240V AC 50/60Hz
Output 12V 7A DC
Plug size: 5.5mm x 2.1 mm, Center Positive

SD Card Slot:
The GreenBox has a micro SD card slot. S32V234 can run applications directly off of the SD card.

Note: we recommend using a Class 10, 16 Gb SD card

UART configuration:
The 'Main UART' is connected to S32V234 through UART0_RXD and UART0_TXD. The provided sample software configures the UART with the following settings:

-speed (baud) 115200
-8 data bits
-1 stop bit
-no parity

S32PDEVPL-KITC: Programing Interface

JTAG

Secondary JTAG

PORT	DESCRIPTION
JTAG	Main JTAG debug interface
Secondary JTAG	Debug interface for daughtercard/ACD

Engine Control Demo

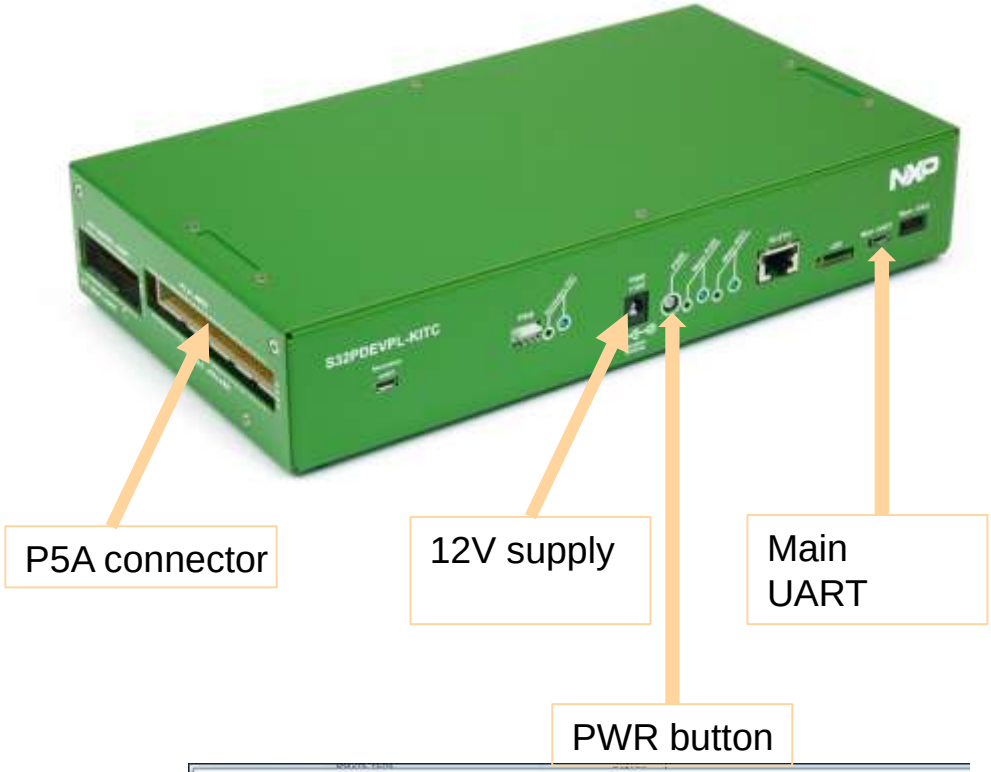
S32PDEVPL-KITC: Engine Management Demo

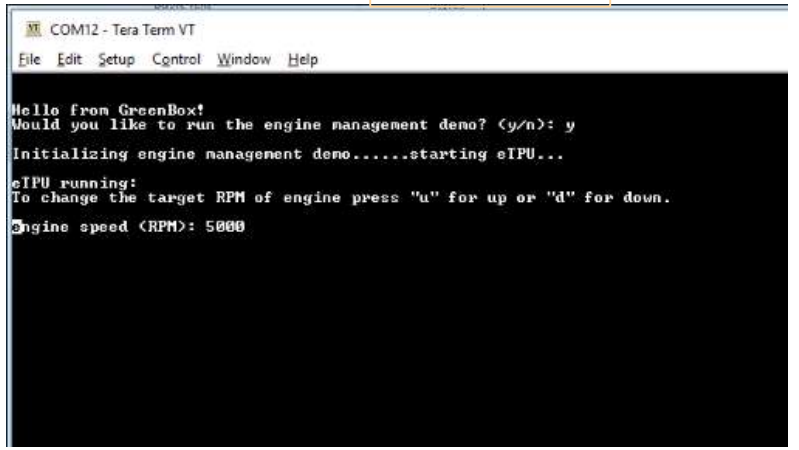
- GreenBox includes engine management software that demonstrates key functions of the eTPU module. This software is based on application note AN4908:
- Engine Control eTPU Demo Application
  - The HW connections and channel assignments used in GreenBox are different from AN4908. Please find the GreenBox hardware pin out on the following slides.
- The application demonstrates the use of software functions that are available in the eTPU engine control library.

eTPU Functions Supported in Demo			
Engine Position	Cam (input) Crank (input) Crank emulator Tooth generator		
Injection	Fuel – cylinder 1 Fuel – cylinder 2 Fuel – cylinder 3 Fuel – cylinder 4	or	Direct injection – bank 1 Direct injection – bank 2 Direct injection – cylinder 1 Direct injection – cylinder 2 Direct injection – cylinder 3 Direct injection – cylinder 4
Ignition	Spark – cylinder 1 Spark – cylinder 2 Spark – cylinder 3 Spark – cylinder 4		

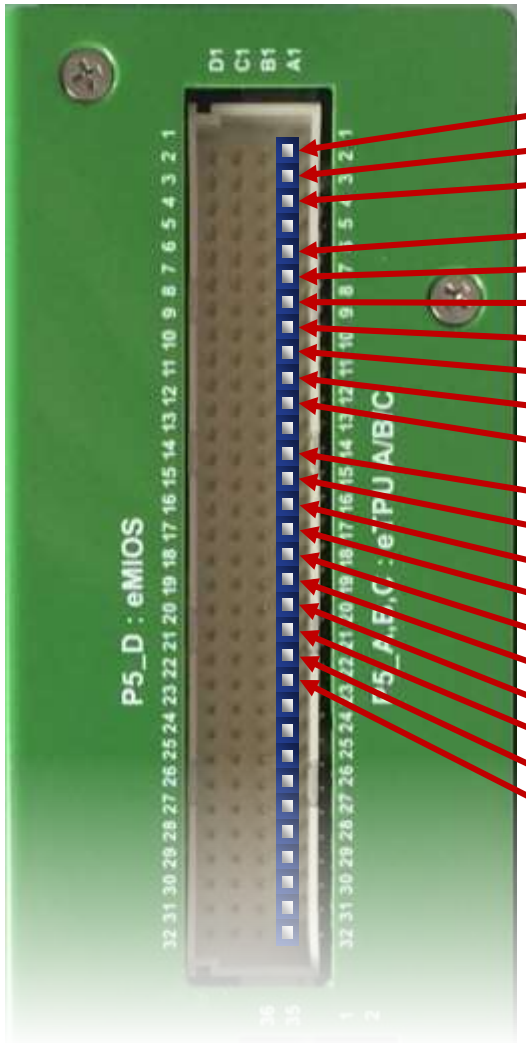
S32SDEVPL-KITC:
Running The Engine Management Demo

- Connect the GreenBox main UART to the PC with the USB cable
  - Set the terminal software (ex: Tera Term) serial baud rate to 115200bps
- Connect eTPU Tooth Generator outputs to eTPU Cam and Crank inputs
  - Connect P5A-02 to P5A-01 (Tooth Gen Cam out to Cam input)
  - Connect P5A-05 to P5A-03 (Tooth Gen Crank to Crank input)
  - See next slide for P5A connector pin out
- Power on the GreenBox
  - Connect the 12 V power supply to the 12 V input on the front middle of the GreenBox
  - Turn on the power by pushing the “PWR” button
- Run the engine management demo
  - The terminal interface will provide an option to run the engine management demo software. Enter "y" from the keyboard.
  - Demo software will run and will display the default value for engine RPM.
  - Change the RPM value by entering "u" (increase RPM) or "d" (decrease RPM) from the keyboard
- Observe the engine control signals (spark ignition and fuel injection)
  - Connect an oscilloscope to the signal pins on the P5A connector (see next slide for P5A pin out)





S32PDEVPL-KITC: Engine Management Demo Connections



Connect/short tooth generator outputs to Cam and Crank Inputs

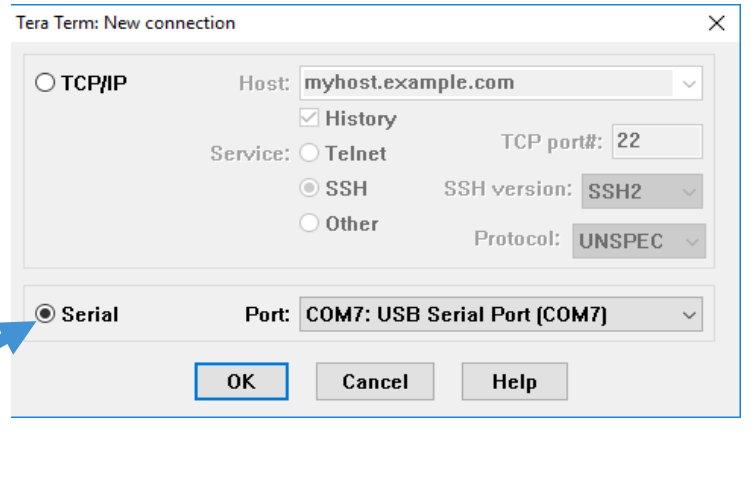
Cam (Input)
Tooth Gen – Cam
Crank (input)
Tooth Gen – Crank
Spark – Cylinder 1
Spark – Cylinder 2
Spark – Cylinder 3
Spark – Cylinder 4
Fuel – Cylinder 1
Fuel – Cylinder 2
Fuel – Cylinder 3
Fuel – Cylinder 4
Knock – Cyl 1 and 4
Knock – Cyl 2 and 3
Direct Inj – Bank 1
Direct Inj – Bank 2
Direct Inj – Cylinder 1
Direct Inj – Cylinder 2
Direct Inj – Cylinder 3
Direct Inj - Cylinder 4

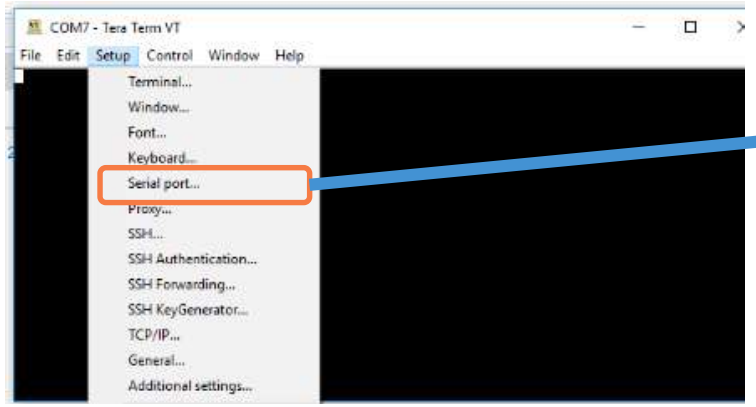
P5A

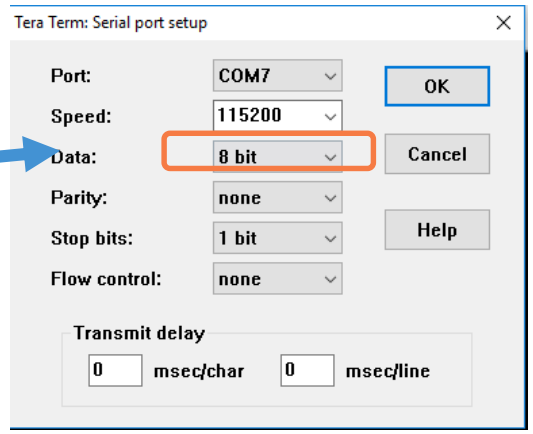
PIN	FUNCTION	PCR#
P5A-01	ETPUA0	114
P5A-02	ETPUA1	115
P5A-03	ETPUA2	116
P5A-04	TP55	
P5A-05	ETPUA4	118
P5A-06	ETPUA5	119
P5A-07	ETPUA6	120
P5A-08	ETPUA7	121
P5A-09	ETPUA8	122
P5A-10	ETPUA9	123
P5A-11	ETPUA10	124
P5A-12	TCRCLKA	113
P5A-13	ETPUA12	126
P5A-14	ETPUA13	127
P5A-15	ETPUA14	128
P5A-16	ETPUA15	129
P5A-17	ETPUA16	130
P5A-18	ETPUA17	131
P5A-19	ETPUA18	132
P5A-20	ETPUA19	133
P5A-21	ETPUA20	134
P5A-22	ETPUA21	135

Console Setup Instructions – (1/2)

- Simple UART control
  - TeraTerm Communication Application
    - Installation Link can be found here.
    - https://osdn.net/projects/ttssh2/releases/
  - Select Serial Connection Type
  - Utilizing Device Manager, determine correct COM port for the GreenBox.
    1. Control Panel → Hardware and Sound → Device Manager
    2. If all drivers installed properly, will be listed under “Ports (COM & LPT)”
  - Set Serial port Speed to **115200**

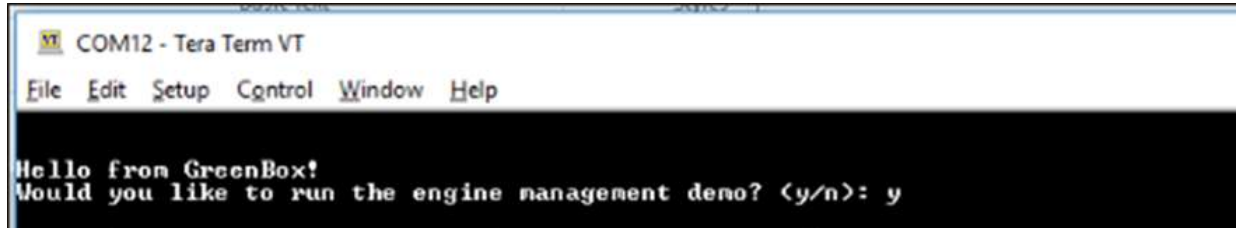


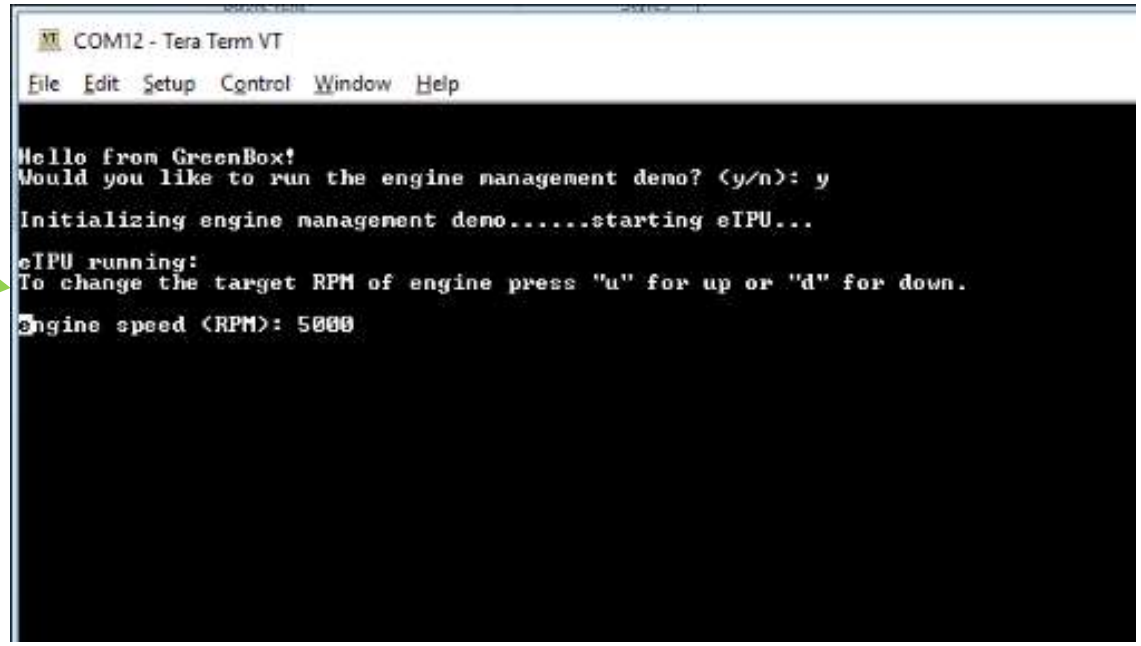




Console Setup Instructions – (2/2)

- If connected properly, power cycle GreenBox, the following should display





- Press “Y” to enable the PWM.
  (press “S” again will disable the PWM)
- Press “u” to increase the RPM
- Press “d” to decrease the RPM

Inverter Control Demo

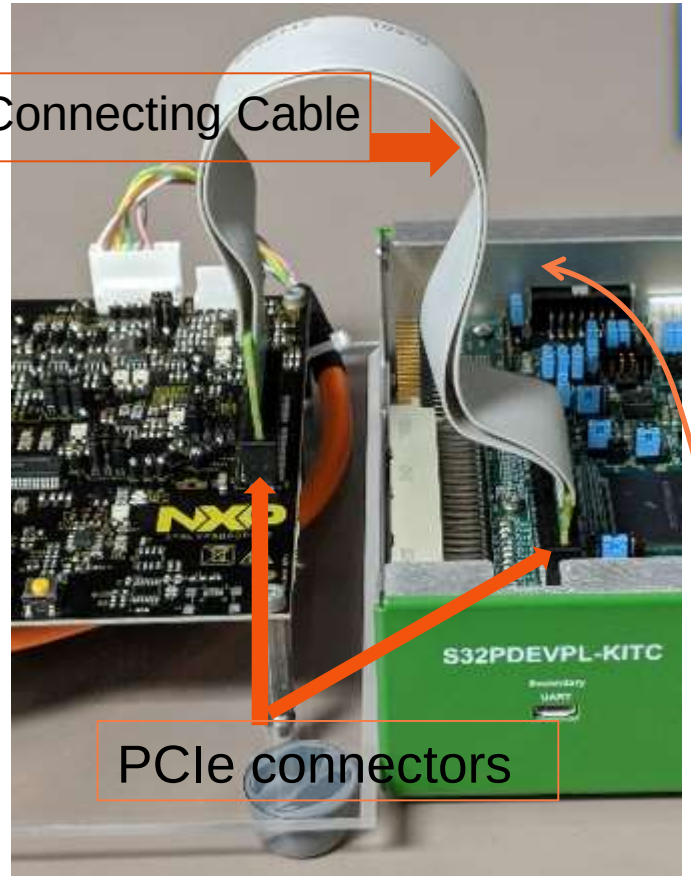
S32PDEVPL-KITC: Inverter Control Demo

- GreenBox includes inverter control demo software that demonstrates 3 phase PMSM (Permanent Magnet Synchronous Motor) motor control using the **eTPU2+**
- Implements the software based resolver using the **eTPU2+**
- Additional information on the PMSM vector control with eTPU can be found in AN2972 application note
- Demo Software is tuned to run on existing NXP hardware solutions. To run the demo the following items are required:
  1. Motor control kit (PN: MTRCKTSPS5744P)
  2. Connecting Cable (PN: X-S32SDEVCON, contact NXP sales)

S32SDEVPL-KITC: Running The Inverter Demo

- Connect the GreenBox main UART to the PC with the USB cable
  - Set the terminal software (ex: Tera Term) serial baud rate to 115200bps
  - See the following slides for more details of the terminal and setup
- Connect eTPU AS trigger to Resolver Sample input
  - Connect P5B-28 to P5B-29 (AS Trigger out to Sample Input)
  - See next slide for P5B connector pin out
- Install the SD Card provided, make sure the software is programmed in the SD as described in “Loading the image onto SD Card” section
- Connect the GreenBox to the Power stage board via a custom Cable
- Power on the GreenBox (12V) /Power stage boards (24V)
  - Connect the 12 V power supply to the 12 V input on the front middle of the GreenBox
  - Turn on the power by pushing the “PWR” button
- Run the inverter control demo
  - The terminal interface will provide an option to run the inverter demo software. Enter “S" or “s” from the keyboard to enable PWM from eTPU
  - Demo software will run and will display the default value for Motor RPM
  - Change the RPM value by entering "u" (increase RPM) or "d" (decrease RPM) from the keyboard

S32SDEVPL-KITC: Power Stage Connection

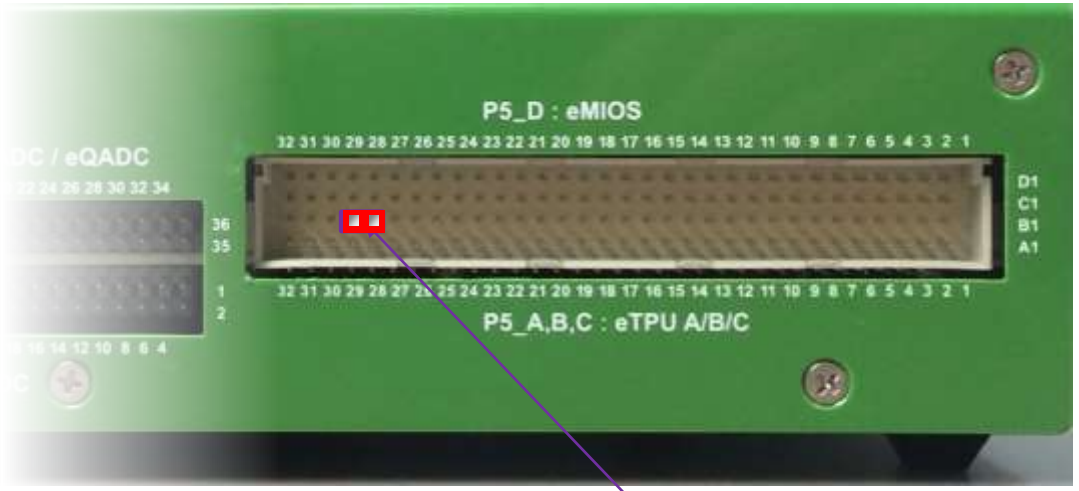


- Connect the GreenBox PCIe connector to power stage PCIe connector via a custom cable. (note do not use any standard PCIe connection cable)
- Install the top lid before powering the board

Remove the Top lid by removing the screws

S32PDEVPL-KITC: Pin Configuration

Side Panel



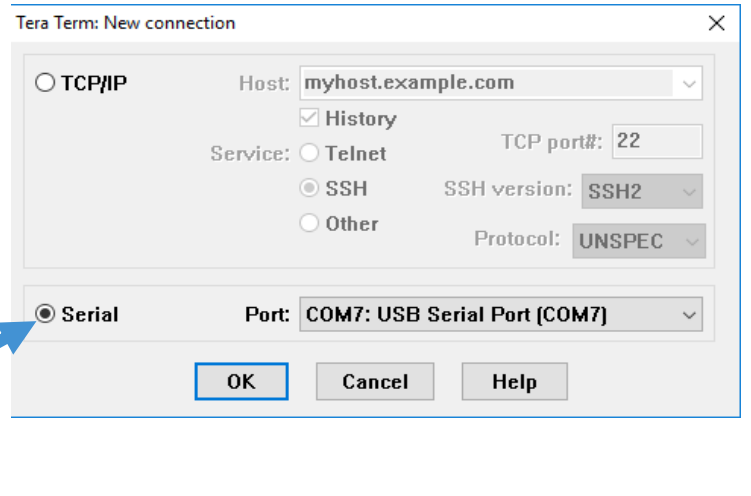
Connect P5B-28 (eTPUB channel 27)
to P5B-29 (eTPUB channel 28)
i.e. Resolver sample input to AS trigger

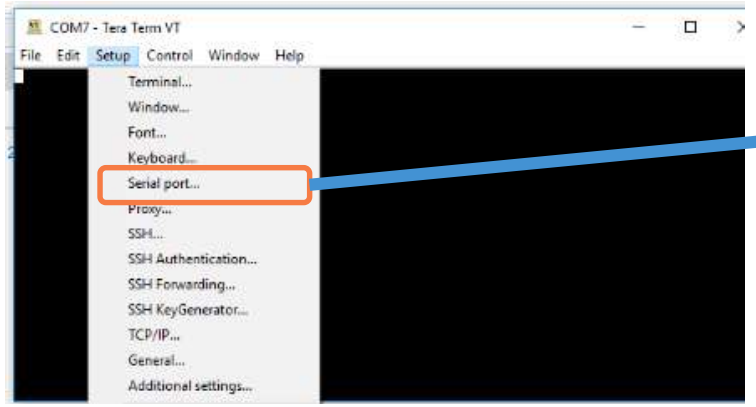
P5B

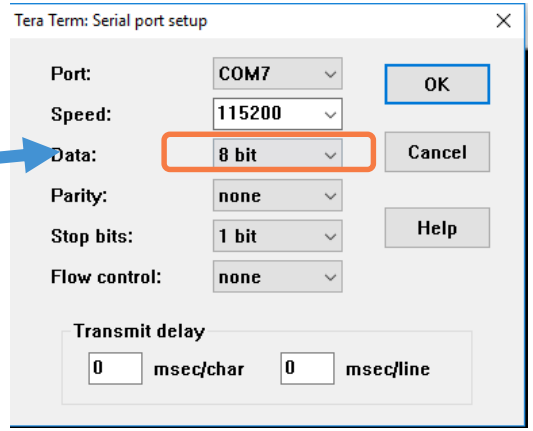
PIN	FUNCTION	PCR#
P5B-01	ETPUB0	147
P5B-02	ETPUB1	148
P5B-03	ETPUB2	149
P5B-04	ETPUB3	150
P5B-05	ETPUB4	151
P5B-06	ETPUB5	152
P5B-07	ETPUB6	153
P5B-08	ETPUB7	154
P5B-09	ETPUB8	155
P5B-10	ETPUB9	156
P5B-11	ETPUB10	157
P5B-12	ETPUB11	158
P5B-13	ETPUB12	159
P5B-14	ETPUB13	160
P5B-15	ETPUB14	161
P5B-16	ETPUB15	162
P5B-17	ETPUB16	163
P5B-18	ETPUB17	164
P5B-19	ETPUB18	165
P5B-20	ETPUB19	166
P5B-21	ETPUB20	167
P5B-22	ETPUB21	168
P5B-23	ETPUB22	169
P5B-24	ETPUB23	170
P5B-25	ETPUB24	171
P5B-26	ETPUB25	172
P5B-27	ETPUB26	173
P5B-28	ETPUB27	174
P5B-29	ETPUB28	175
P5B-30	ETPUB29	176
P5B-31	ETPUB30	177
P5B-32	ETPUB31	178

Console Setup Instructions – (1/2)

- Simple UART control
  - TeraTerm Communication Application
    - Installation Link can be found here.
    - https://osdn.net/projects/ttssh2/releases/
  - Select Serial Connection Type
  - Utilizing Device Manager, determine correct COM port for the GreenBox.
    1. Control Panel → Hardware and Sound → Device Manager
    2. If all drivers installed properly, will be listed under “Ports (COM & LPT)”
  - Set Serial port Speed to **115200**

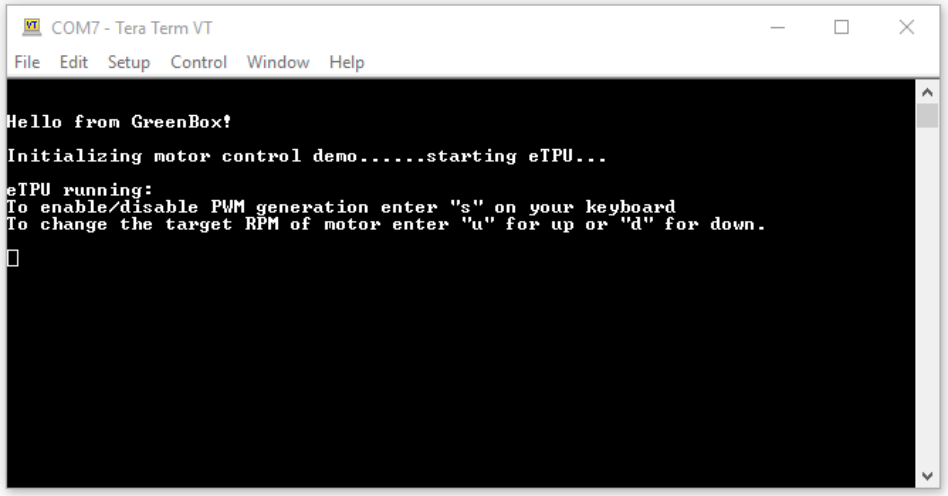




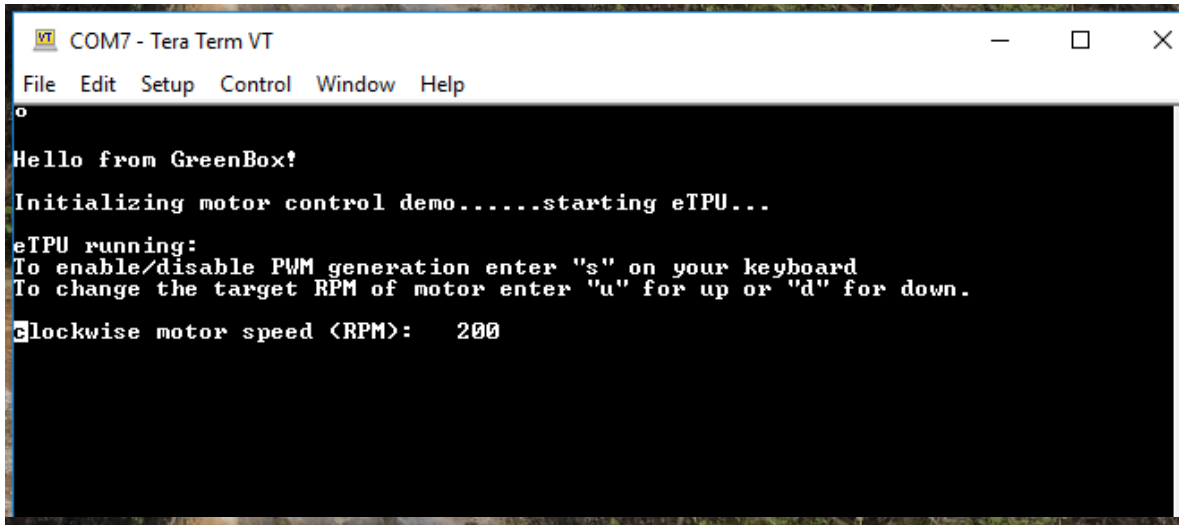


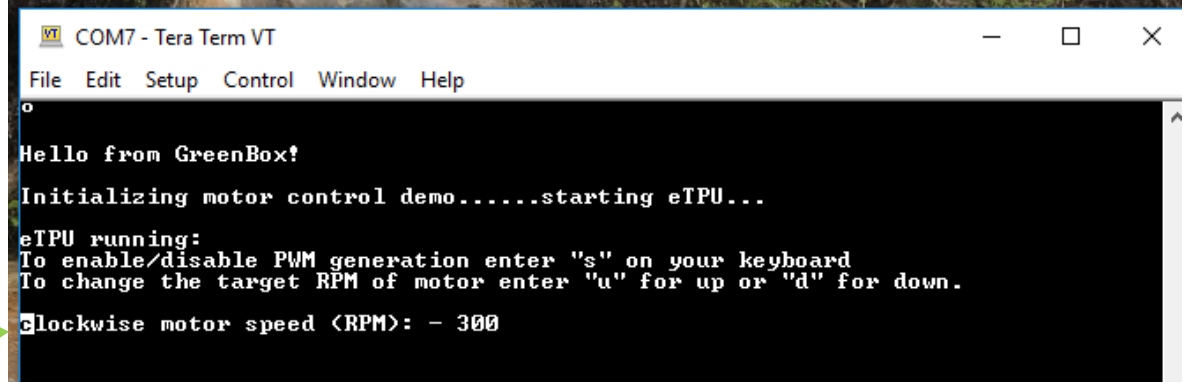
Console Setup Instructions – (2/2)

- If connected properly, power cycle GreenBox, the following should display



- Press “S” to enable the PWM.
  (press “S” again will disable the PWM)
- Press “u” to increase the RPM
- Press “d” to decrease the RPM





Software

GreenBox Software

AUTOSAR operating system

- ARM A53 cores

MCAL drivers

- S32V234 peripheral support

Zipwire interface

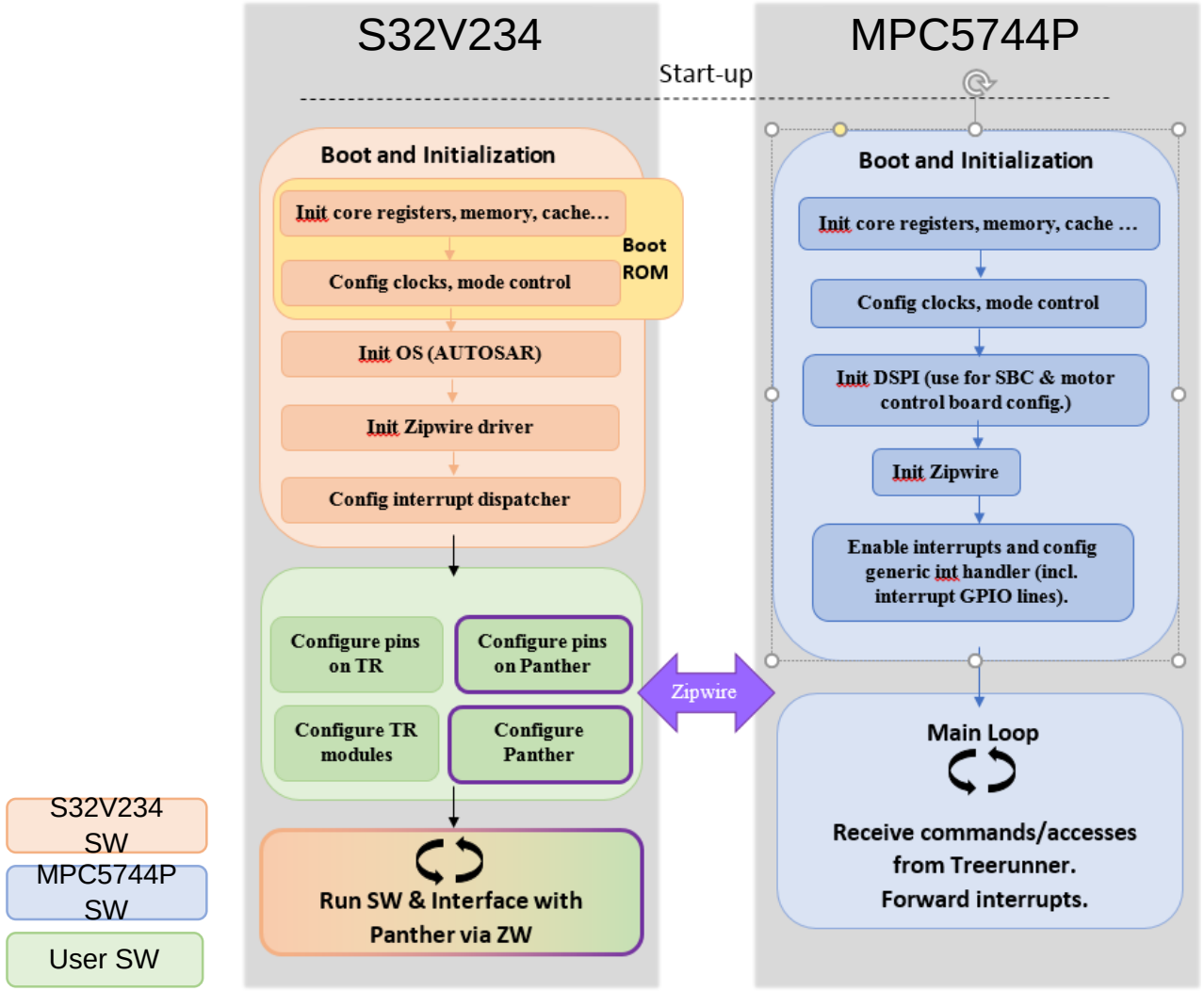
- MPC5744P peripheral access/control
  - Register R/W
  - Abstracts peripheral access for sw application

Development Tools

- S32 Design Studio, VSDK for S32V234

Demo Software

- FOC PMSM motor control
- Combustion engine management
- Base demo project



GreenBox Software

- GreenBox software can be downloaded at www.nxp.com/GreenBox
  Note: software can only be downloaded from the site by users who have received a GreenBox.

1. GreenBox Base project
   - This is the baseline software framework provided for the GreenBox development platform

2. GreenBox ICE (eTPU2+ based **I**nternal **C**ombustion **E**ngine) demo software

3. GreenBox Inverter (eTPU2+ based PMSM control with software resolver) demo software (PMSM: **P**ermanent **M**agnet **S**ynchronous **M**otor)

Step-by-Step Import and Debug With S32 Design Studio

The **S32PDEVPL-KITC** Quick Start Guide takes you step by step through creating, running, and debugging projects from Internal S32V SRAM, External DDR, or from an SD Card. (S32V does not contain any Internal Flash)

1

Install Software and Tools

Install S32 Design Studio IDE for Vision
S32 Design Studio for Vision

2

Connect the Power Adapter and Debug Interface

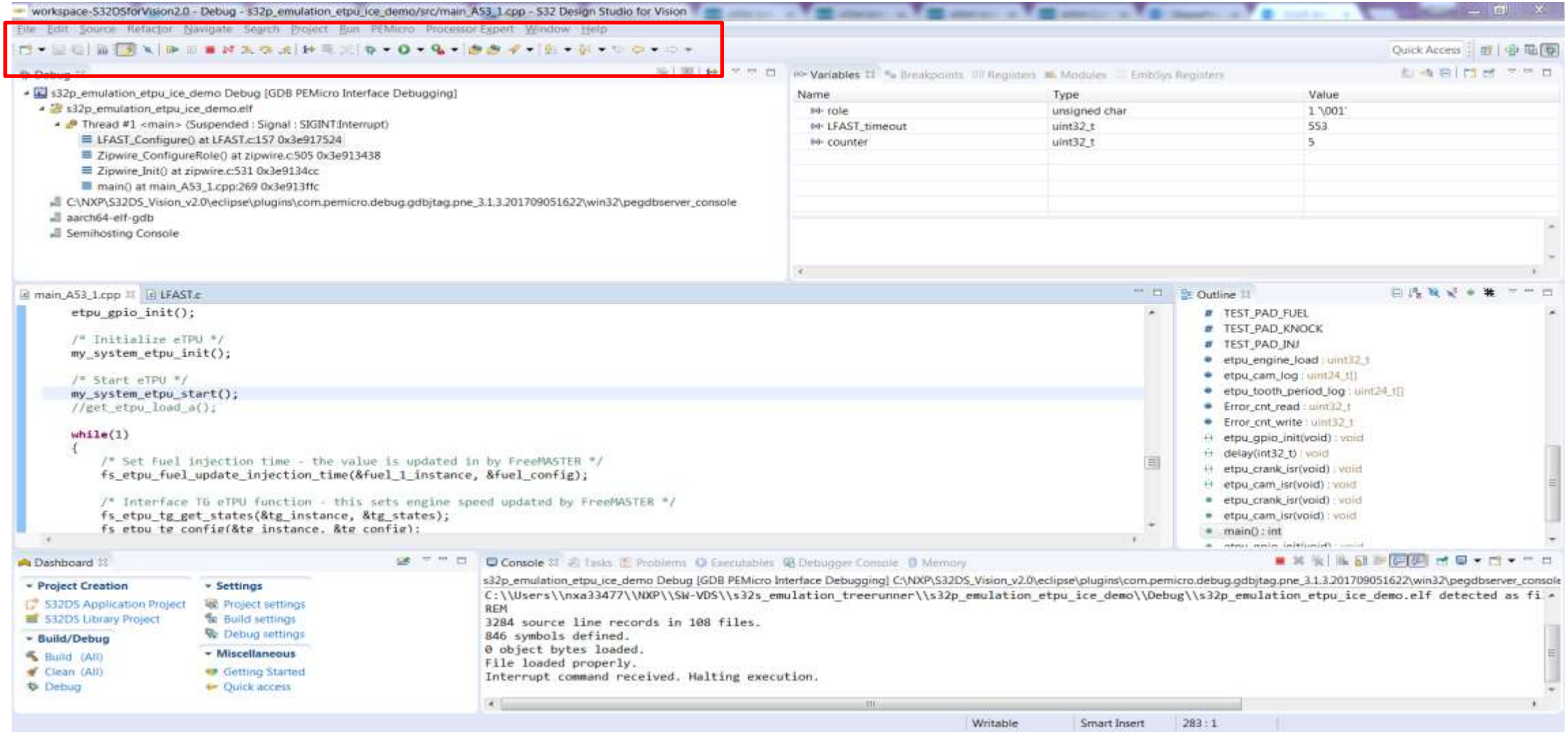
Connect the 12 V power supply to the 12 V input on the front middle of the GreenBox. Additionally, connect the PE Micro Debug Interface to the main JTAG connector on the front right side of the GreenBox using a 10-pin JTAG connector.

3

Launch S32 Design Studio

Launch S32 Design Studio and follow the steps on the following slides to import, compile, and debug a project.

S32 Design Studio Project IDE & Debugger (Eclipse & GCC)



S32PDEVPL-KITC: Zipwire Driver

- The Zipwire driver allows S32V234 to communicate with the peripherals on MPC5777C
- To use the driver include **Zipwire_API.h** into your code.
- The API contains the following functions:

Zipwire Driver Functions
uint8_t **Zipwire_MCAL_Read8**(uint32_t address);
uint16_t **Zipwire_MCAL_Read16**(uint32_t address);
uint32_t **Zipwire_MCAL_Read32**(uint32_t address);
void **Zipwire_MCAL_Write8**(uint32_t address, uint8_t value);
void **Zipwire_MCAL_Write16**(uint32_t address, uint16_t value);
void **Zipwire_MCAL_Write32**(uint32_t address, uint32_t value);
uint8_t **Zipwire_Reset**(void);
uint8_t **Zipwire_ConfigureRole**(uint8_t role);
uint8_t **Zipwire_InitChannel**(uint8_t channelNo);
uint8_t **Zipwire_Init**(uint8_t role, uint8_t channel);

Zipwire Driver Files
LFAST.c/.h
SIPI_API.h
SIPI_HSSL_API.c
SIPI_HSSL_Header_v4.h
Zipwire.c/.h
Zipwire_API.h

Note: the above files must be included in the project

S32PDEVPL-KITC: Zipwire Driver

- To initialize Zipwire you must call the **Zipwire_Init** function

  Zipwire_Init(ZIPWIRE_MASTER_NODE, ZIPWIRE_MCAL_CHANNEL)

  - Note: the initialization of the slave is done with the firmware pre-flashed on MPC5777C
- Upon successful initialization, the read and write functions can be used to transfer data. For more details of each function please see the **Zipwire_API.h** header file.

Please note that in the eTPU sample application, the Zipwire API read/write functions are not called directly. Instead **MemAccessMacros.h** is included so that the following list of read/write functions can be used instead. These macros provide a more generic interface that either calls the Zipwire read/write function or do a traditional memory access depending on if USE_ZIPWIRE is defined within the code.

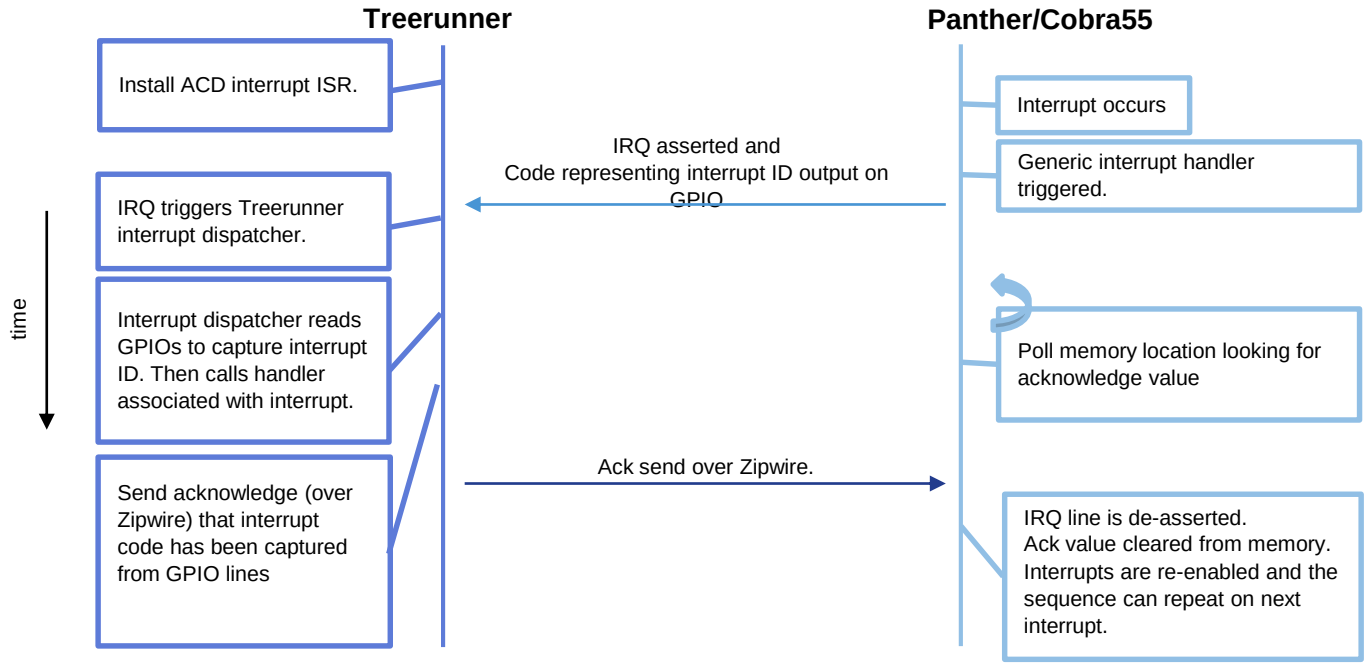
REG_WRITE8(address, value)
REG_WRITE16(address, value)
REG_WRITE32(address, value)
REG_READ8(address)
REG_READ16(address)
REG_READ32(address)

Interrupt Processing and Software Requirements

S32P Emulation platform is based on S32V234 main processor and SPC5777C slave processor (ACD) to emulate the next generation S32Px processor with latest NXP Multi-chip solution. All the ACD interrupts will be forwarded to the S32V234 via GPIO and a IRQ to the main processor. The main processor software decodes the relevant ACD interrupt and jumps to the User defined interrupt service routine. After the user ISR executes an interrupt acknowledge will be sent to ACD via Zipwire (this is handled by the ACD interrupt dispatcher driver).

To handle the interrupts forwarded from the modules on the slave processor, the user must follow these steps in the bare metal software project.

1) Define the interrupt service routine (ISR) for the desired interrupt, ex: ISR_eMIOS_0_channel_0_flag()
2) Install the ACD interrupt ISR referring the lookup table “code” and ISR name shown in following example.
   interrupt_dispatcher_InstallInterruptHandler(ISR_eMIOS_0_channel_0_flag , 0x1); //0x1 : Code for the forwarded ACD interrupt
3) The hand shake shown in following diagram is handled by the ACD interrupt dispatcher driver



S32PDEVPL-KITC: Default Configuration of MPC5777C

- The MPC5777C inside the GreenBox comes pre-flashed with firmware
- The included firmware does the following
  - Boots from flash
  - Performs basic MCU initialization
  - Initializes the clocks
  - Initializes Zipwire interface as slave device
  - Initializes the interrupt forwarding service

MPC5777C Clock Configuration Details
PLL0 to 200MHz (40MHz XOSC reference)
PLL1 to 262.5MHz (50MHz PLL0 PHI1 output reference)
SYSCLKSEL = PLL1
PERCLKSEL = PLL0
SYSCLKDIV = Divide-by-1 (SYSCLK is 262.5 MHz)
PERDIV = Divide-by-2 (PER CLK is 100 MHz)
FMPERDIV = Divide-by-2 (FMPER CLK is 131.25 MHz)
ETPUDIV = Divide-by-1

What To Expect From The GreenBox

- GreenBox Demos run the S32V A53 Cores at 800MHz
  - This is the Target frequency for the R52 Cores on the S32S,E,P devices.
- A53's run the 64 Bit Instruction Set by default
  - Switching to the 32 Instruction Set is not trivial.
  - GreenBox runs the 64 Bit Instruction Set.
- Performance of strait C code is about 3X that of Cobra 55 at 264MHz
  - This is per A53 Core, and there are 4 Cores available.
- Expected performance of the S32x Family with R52 Cores:
  - About 4X Cobra55 at 264MHz per Core, and there will be 4 Lockstep Cores available
  - Boost in performance will be from:
    - Tightly Coupled SRAM, 32 Bit Instruction Set, Real Time Interrupt Processing, etc.

Documentation and Reference Material

- Documentation Links
  - S32V234 Reference Manual
  - S32V234 Data Sheet
  - MPC577C Reference Manual
  - MPC5777C Data Sheet

- Quick Start Guide
  - S32PDEVPL-KITC_QSG_V3.1_SP.PPTX (see your local NXP FAE or Sales Person)

- Application Notes
  - AN4907: Engine Control eTPU Library
  - AN4908: Engine Control eTPU Demo Application
  - AN5374: eTPU library usage in an application
  - AN2972: Using the PMSM Vector Control eTPU Function
  - ARM Cortex-A Programmer’s guide for ARMv8-A
  - ARM Cortex-A53 Technical Reference Manual

SECURE CONNECTIONS
FOR A SMARTER WORLD

www.nxp.com