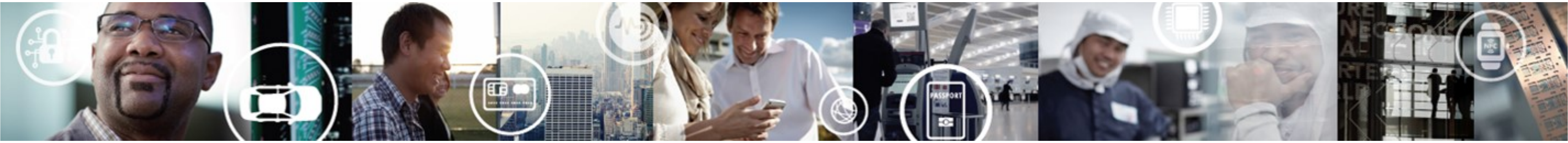
MPC5748G-GW-RDB
EXAMPLE CODES USER GUIDE (ECUG)

Ultra-Reliable MCUs for Industrial and Automotive Applications



SECURE CONNECTIONS
FOR A SMARTER WORLD

SUMMARY

- 1. Hands-on - CAN
- 2. Hands-on - CAN_FD
- 3. Hands-on - ENET0+SPI
- 4. Hands-on - ENET1
- 5. Hands-on - UART
- 6. Hands-on - LIN

01.

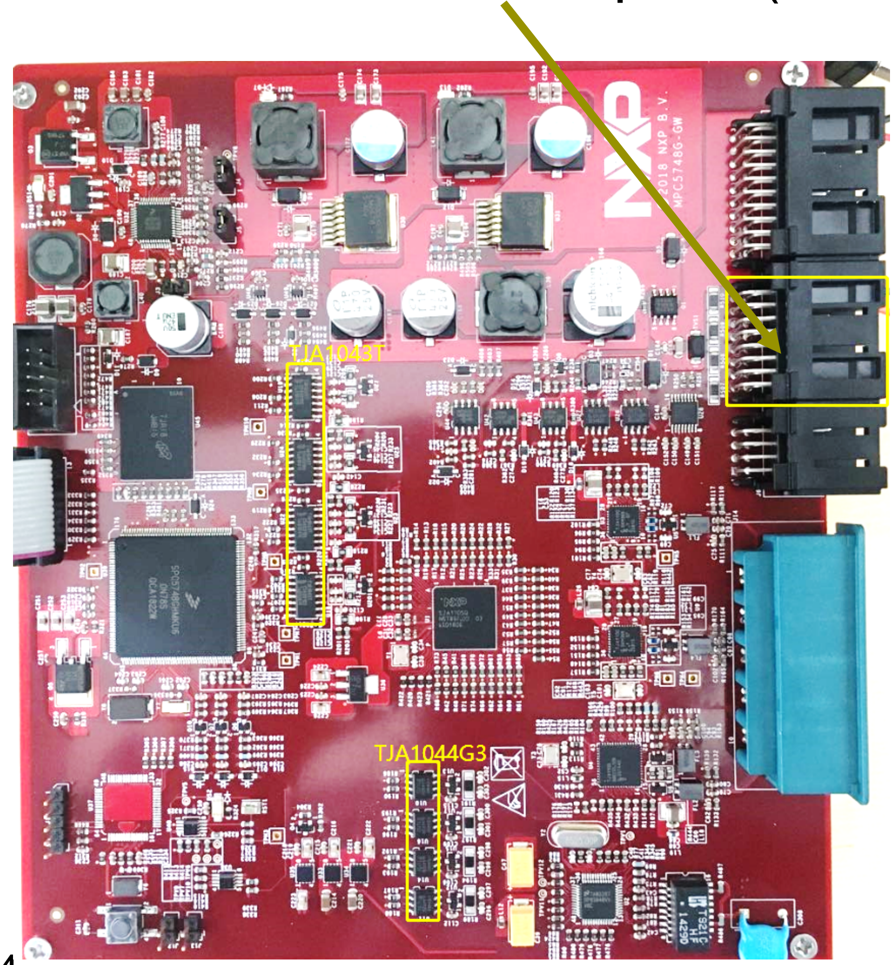
Hands-on – CAN

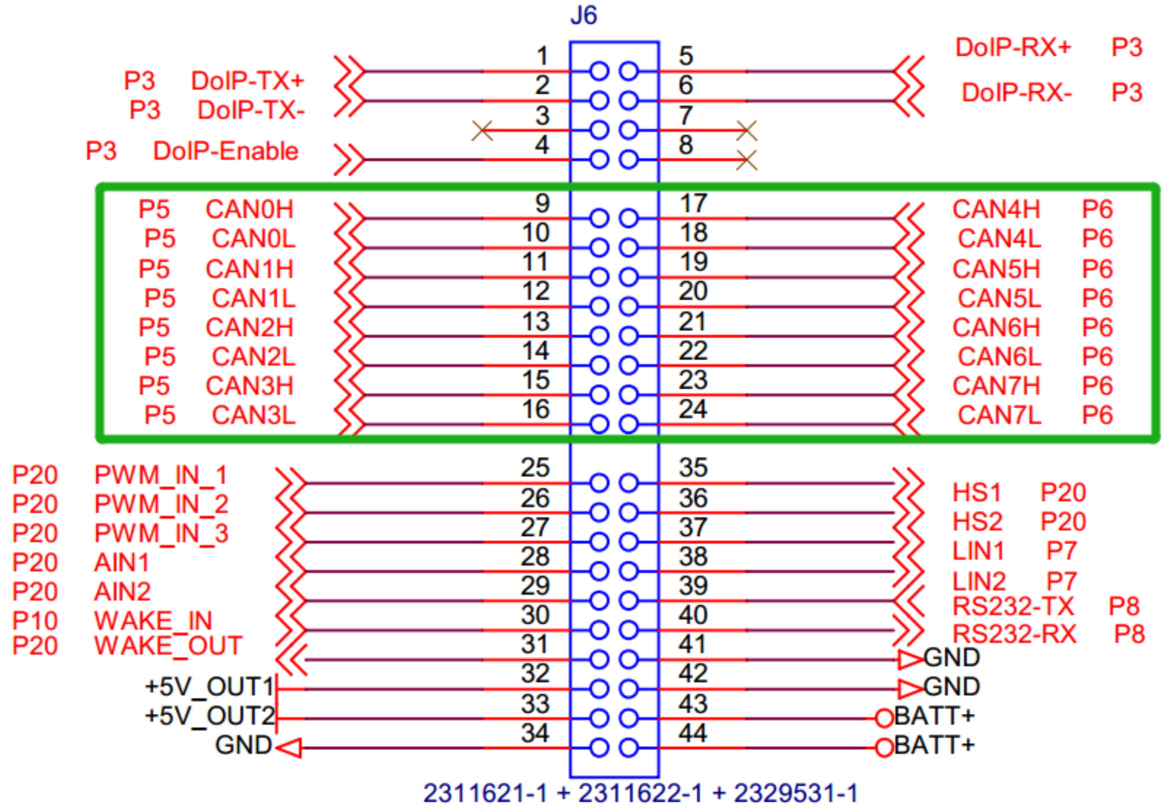
Hands-on – CAN: Objective

- Features of FlexCAN module on MPC5748G
- How to set a pin as output/input with SDK
- How to configure the port of CAN
- How to modify an existing SDK project with S32DS to suit this board
- Use CAN0~CAN7 to send CAN message
- Use CAN0~CAN7 to receive CAN message

Hands-on – CAN: Resources

- Resources to be used:
  - On-board user CAN ports (hardwired to GPIOs)

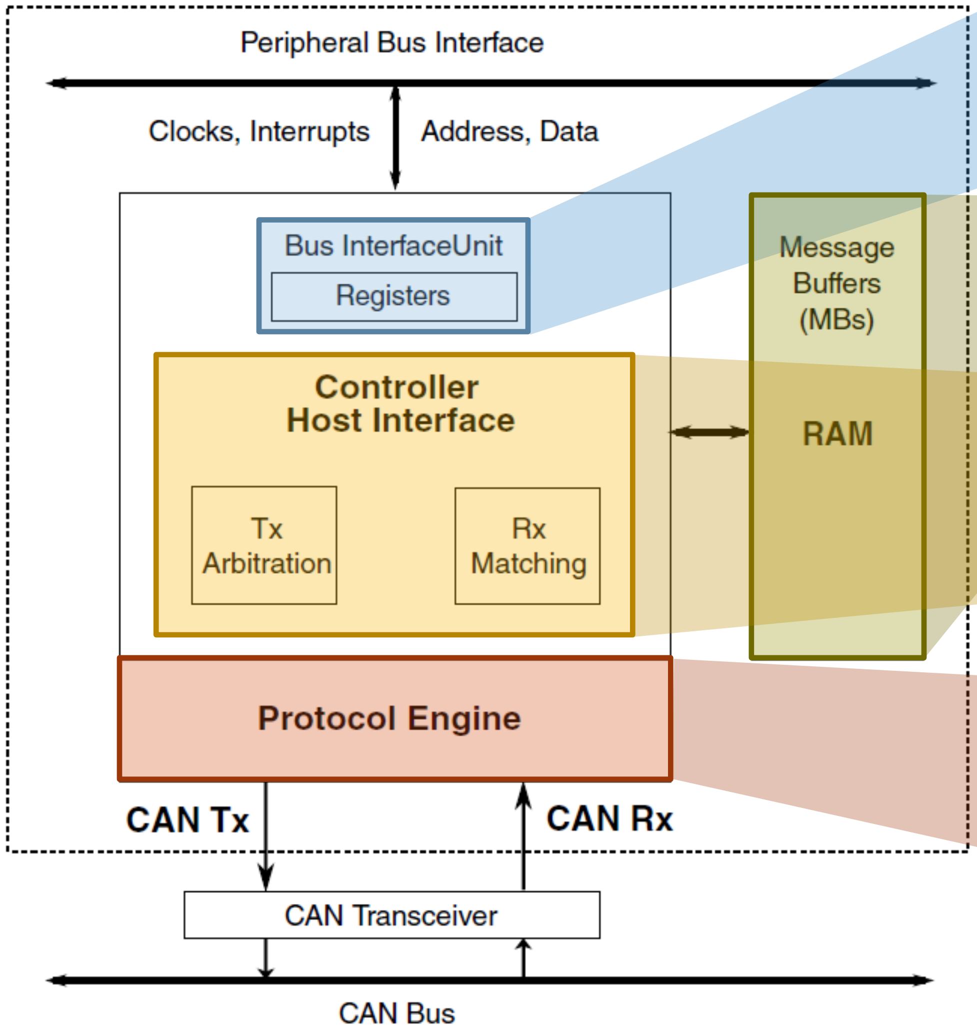




Hands-on – CAN: Theory

- Full implementation of the CAN FD & CAN 2.0 B
  - data field bitrate up to 8Mbps
- Flexible mailboxes (0/8/16/32/64 bytes data length)
- Listen-Only mode capability
- Programmable Loop-Back mode supporting self-test operation
- Programmable transmission priority scheme
- Independence from the transmission medium
- CRC status for transmitted message
- Full featured Rx FIFO with storage capacity for 6 frames
- DMA request for Rx FIFO
- Programmable clock source to the CAN Protocol Interface, either bus clock or crystal oscillator
- 100% backward compatibility with previous FlexCAN version
- 8 FlexCAN instances

Hands-on – CAN: Theory



Access to and from the internal interface bus (clocks, address and data buses, interrupts, DMA and test signals)

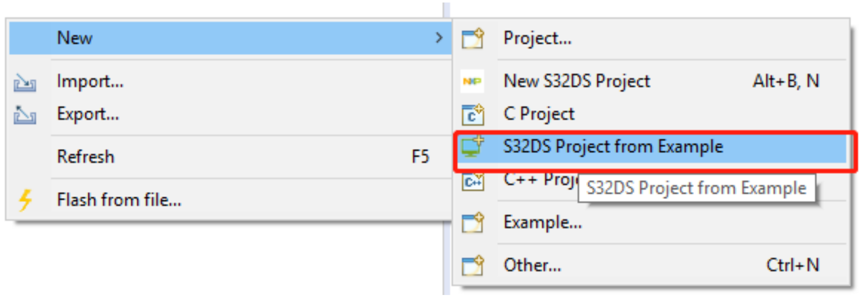
Embedded RAM dedicated to the FlexCAN

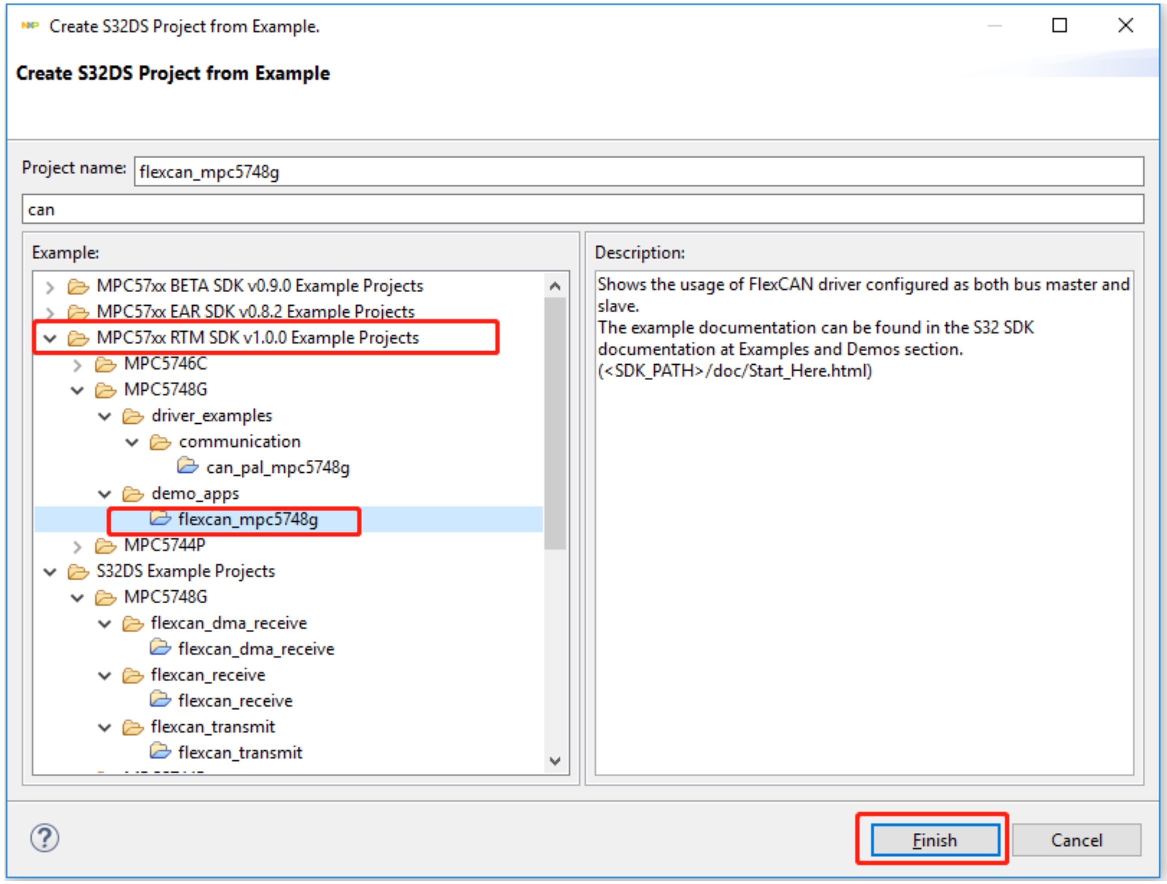
Message buffer selection for reception and transmission (arbitration and ID matching algorithms)

Serial communication on the CAN bus (RAM access requests for rx and tx frames, rx messages validation, error handling)

Hands-on – CAN: Import Existing Project

- Create S32DS project from Example:
  - File->New->New S32DS Project from Example
  - Select: **flexcan_mpc5748g** from **MPC57xxRTM SDK v1.0.0 Example Projects**





Project name: flexcan_mpc5748g

can

Shows the usage of FlexCAN driver configured as both bus master and slave.
The example documentation can be found in the S32 SDK documentation at Examples and Demos section.
(<SDK_PATH>/doc/Start_Here.html)

Hands-on – CAN: Modify

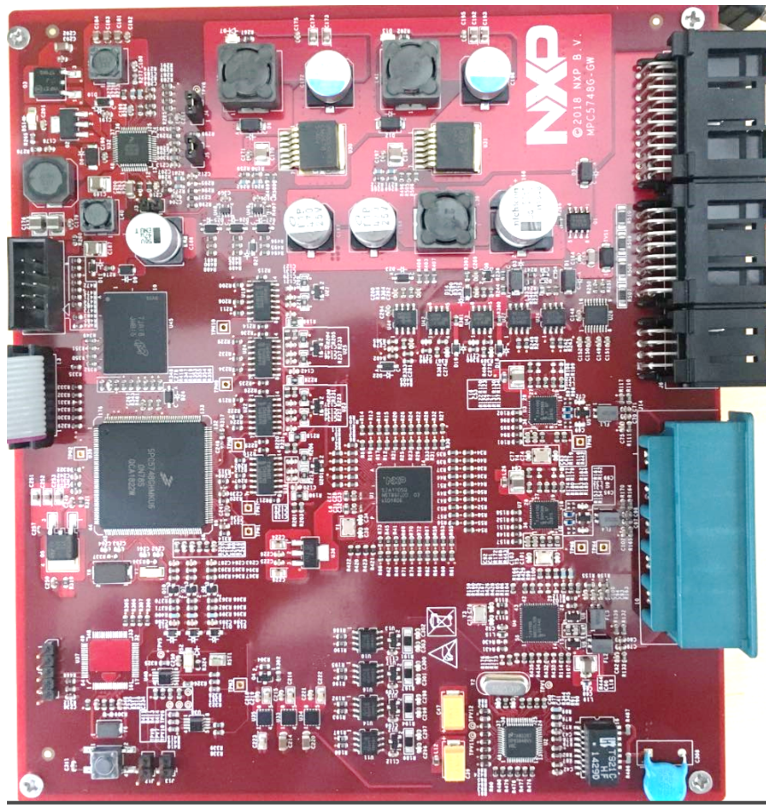
- The example of flexcan_mpc5748g project is suit to DEVKIT
- How to modify?

DEVKIT

Modify

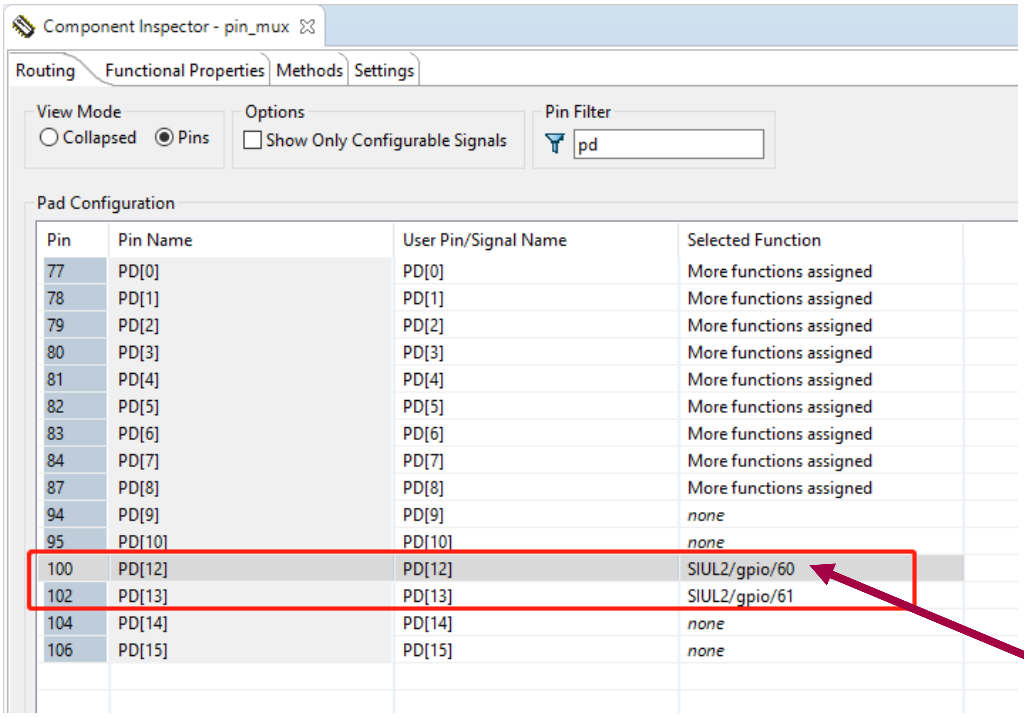
Jobs

1.Peripheral Power Supply
2.CAN Phy enablement
3.The Ports of CAN
4.CAN configuration
5.Application code of the main.c



Hands-on – CAN: Modify-Peripheral Power Supply

- Enable the peripheral power supply
- Open ‘pin_mux’ component in ‘Component Inspector’ to configure pin routing
- SIUL2 tab -> GPIO 60（61） > select the pin (one option) + direction output

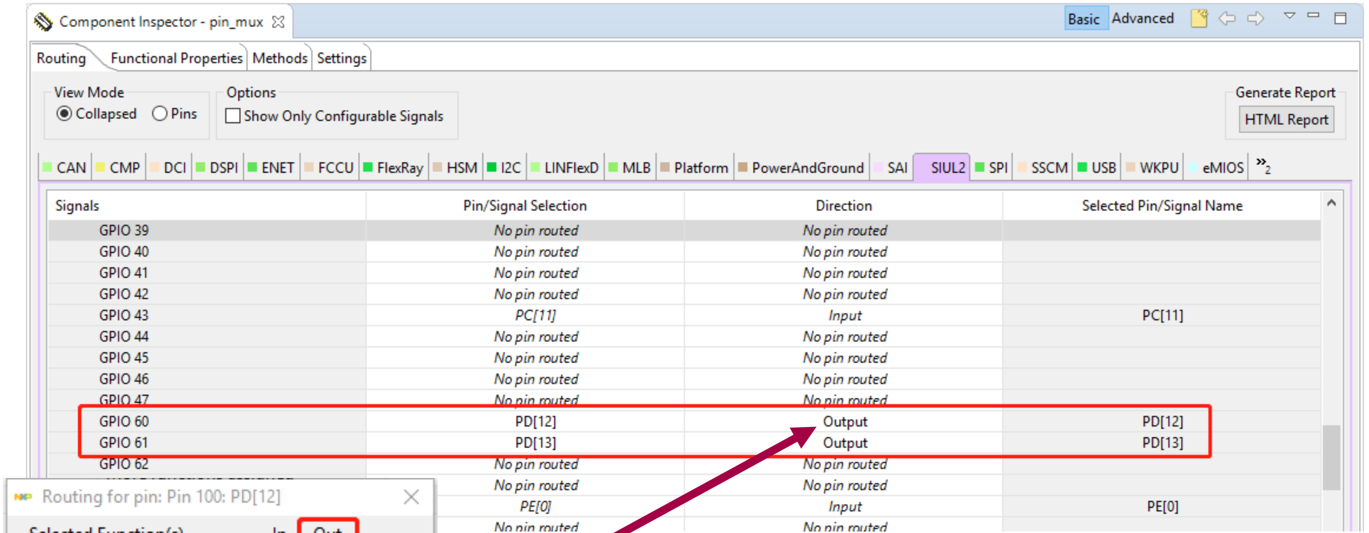

Component Inspector - pin_mux

Routing | Functional Properties | Methods | Settings

View Mode: ( ) Collapsed [✓ Pins]
Options: ☐ Show Only Configurable Signals
Pin Filter: pd

Pad Configuration

Pin	Pin Name	User Pin/Signal Name	Selected Function
77	PD[0]	PD[0]	More functions assigned
78	PD[1]	PD[1]	More functions assigned
79	PD[2]	PD[2]	More functions assigned
80	PD[3]	PD[3]	More functions assigned
81	PD[4]	PD[4]	More functions assigned
82	PD[5]	PD[5]	More functions assigned
83	PD[6]	PD[6]	More functions assigned
84	PD[7]	PD[7]	More functions assigned
87	PD[8]	PD[8]	More functions assigned
94	PD[9]	PD[9]	*none*
95	PD[10]	PD[10]	*none*
100	PD[12]	PD[12]	SIUL2/gpio/60
102	PD[13]	PD[13]	SIUL2/gpio/61
104	PD[14]	PD[14]	*none*
106	PD[15]	PD[15]	*none*


Component Inspector - pin_mux

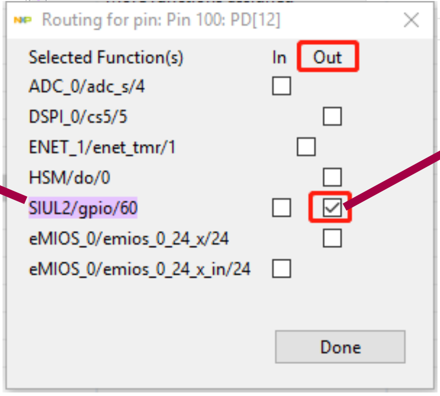
Basic | Advanced

Routing | Functional Properties | Methods | Settings

View Mode: [✓ Collapsed] ( ) Pins
Options: ☐ Show Only Configurable Signals
Generate Report: HTML Report

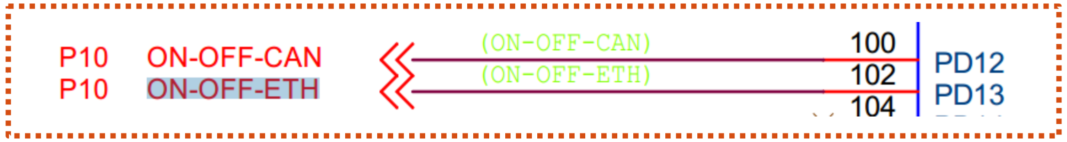
CAN | CMP | DCI | DSPI | ENET | FCCU | FlexRay | HSM | I2C | LINFlexD | MLB | Platform | PowerAndGround | SAI | SIUL2 | SPI | SSCM | USB | WKPU | eMIOS

Signals	Pin/Signal Selection	Direction	Selected Pin/Signal Name
GPIO 39	*No pin routed*	*No pin routed*	
GPIO 40	*No pin routed*	*No pin routed*	
GPIO 41	*No pin routed*	*No pin routed*	
GPIO 42	*No pin routed*	*No pin routed*	
GPIO 43	*PC[11]*	*Input*	PC[11]
GPIO 44	*No pin routed*	*No pin routed*	
GPIO 45	*No pin routed*	*No pin routed*	
GPIO 46	*No pin routed*	*No pin routed*	
GPIO 47	*No pin routed*	*No pin routed*	
GPIO 60	PD[12]	Output	PD[12]
GPIO 61	PD[13]	Output	PD[13]
GPIO 62	*No pin routed*	*No pin routed*	
	No pin routed	*No pin routed*	
	PE[0]	*Input*	PE[0]
	No pin routed	*No pin routed*	


Routing for pin: Pin 100: PD[12]

Selected Function(s)	In	Out
ADC_0/adc_s/4	☐	
DSPI_0/cs5/5		☐
ENET_1/enet_tmr/1	☐	
HSM/do/0		☐
SIUL2/gpio/60	☐	☑
eMIOS_0/emios_0_24_x/24		☐
eMIOS_0/emios_0_24_x_in/24	☐	

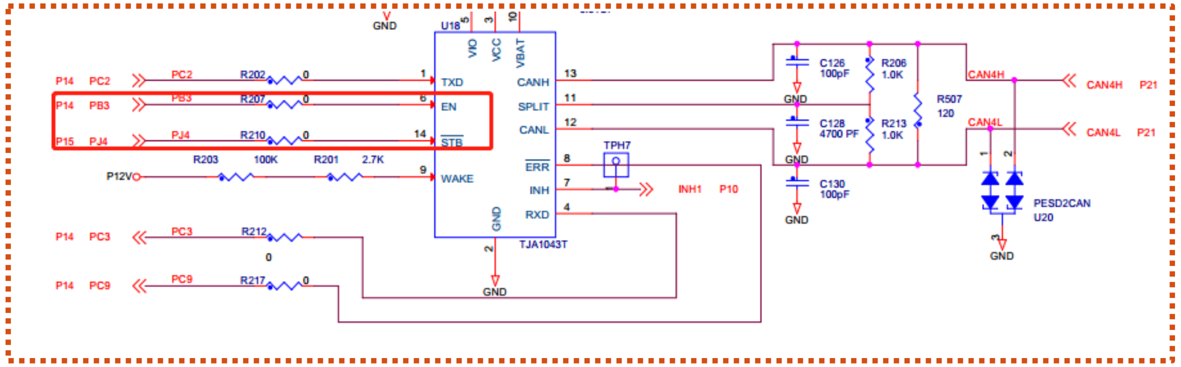
Done

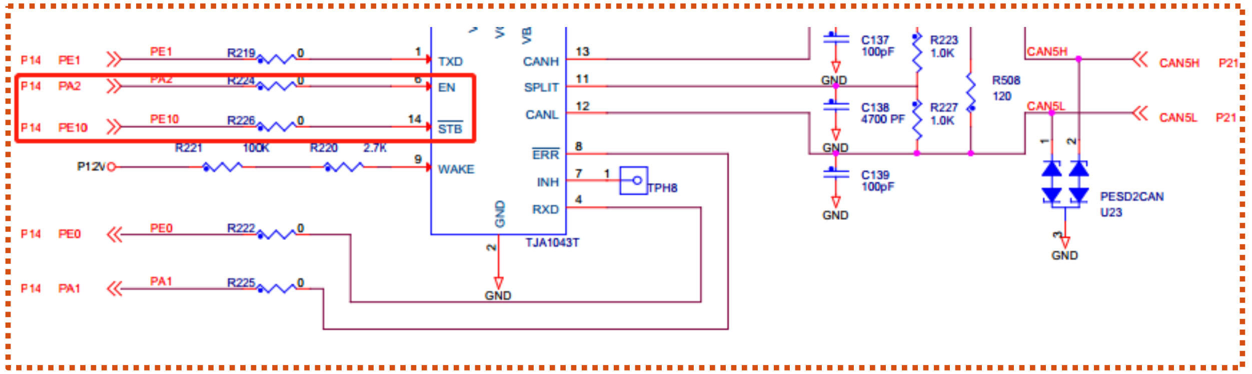


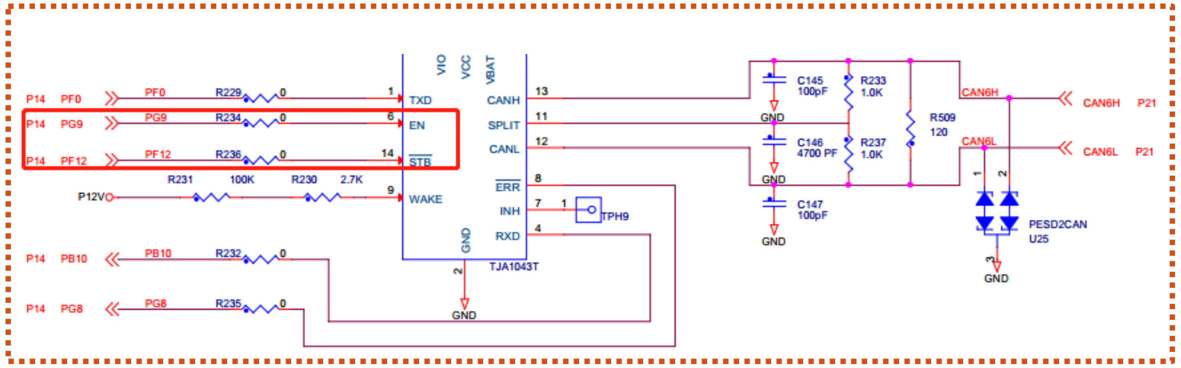
```
void Peripheral_Power_Supply_Init(void)
{
    PINS_DRV_WritePin(PTD, 12, 1);
    PINS_DRV_WritePin(PTD, 13, 1);
}
```

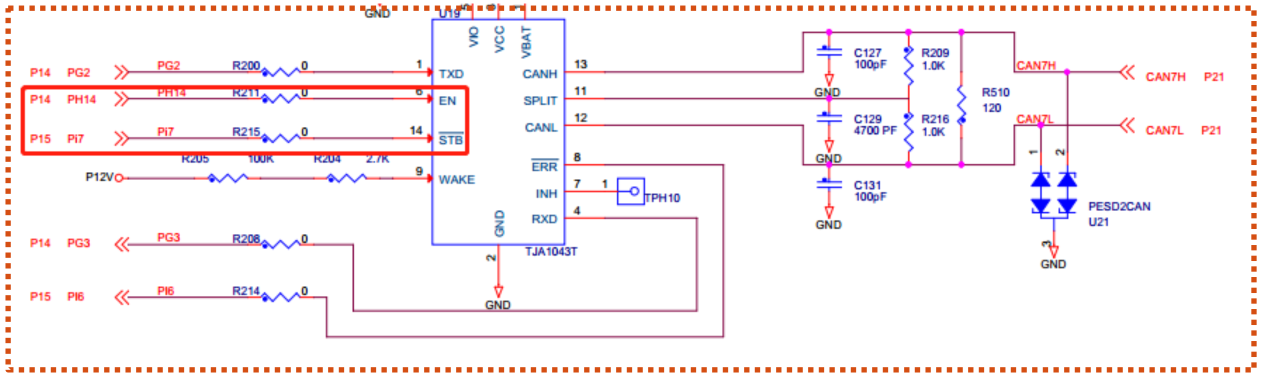
Hands-on – CAN: Modify-CAN Phy Enablement

- Configure the CAN4~7 Phy(TJA1043T) GPIO





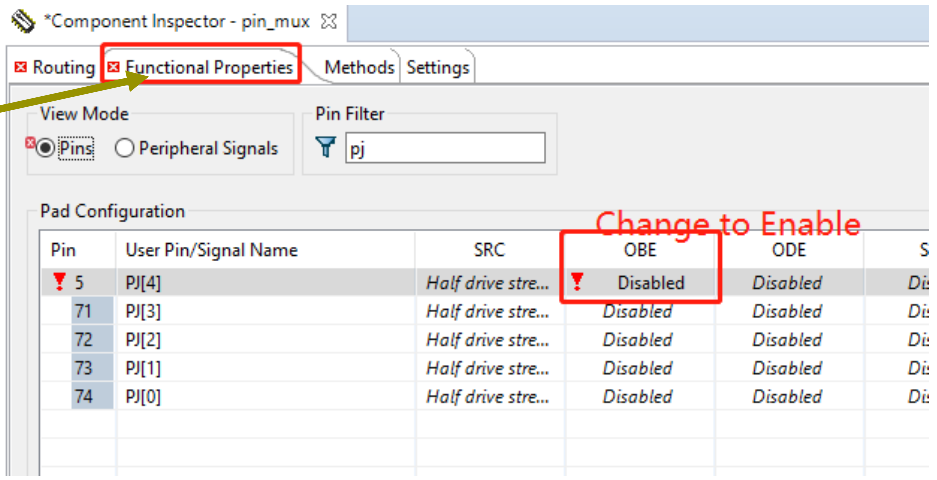




Hands-on – CAN: Modify-CAN Phy Enablement

- The configuration method is similar to the slides on the previous page

CAN Phy	port	routing	direction	Note
CAN4 EN	PB[3]	SIUL2/gpio/19	Out	
CAN4 STB	PJ[4]	SIUL2/gpio/148	Out	There is a problem
CAN5 EN	PA[2]	SIUL2/gpio/2	Out	
CAN5 STB	PE[10]	SIUL2/gpio/74	Out	
CAN6 EN	PG[9]	SIUL2/gpio/105	Out	
CAN6 STB	PF[12]	SIUL2/gpio/92	Out	
CAN7 EN	PH[14]	SIUL2/gpio/126	Out	
CAN7 STB	PI[7]	SIUL2/gpio/135	Out	



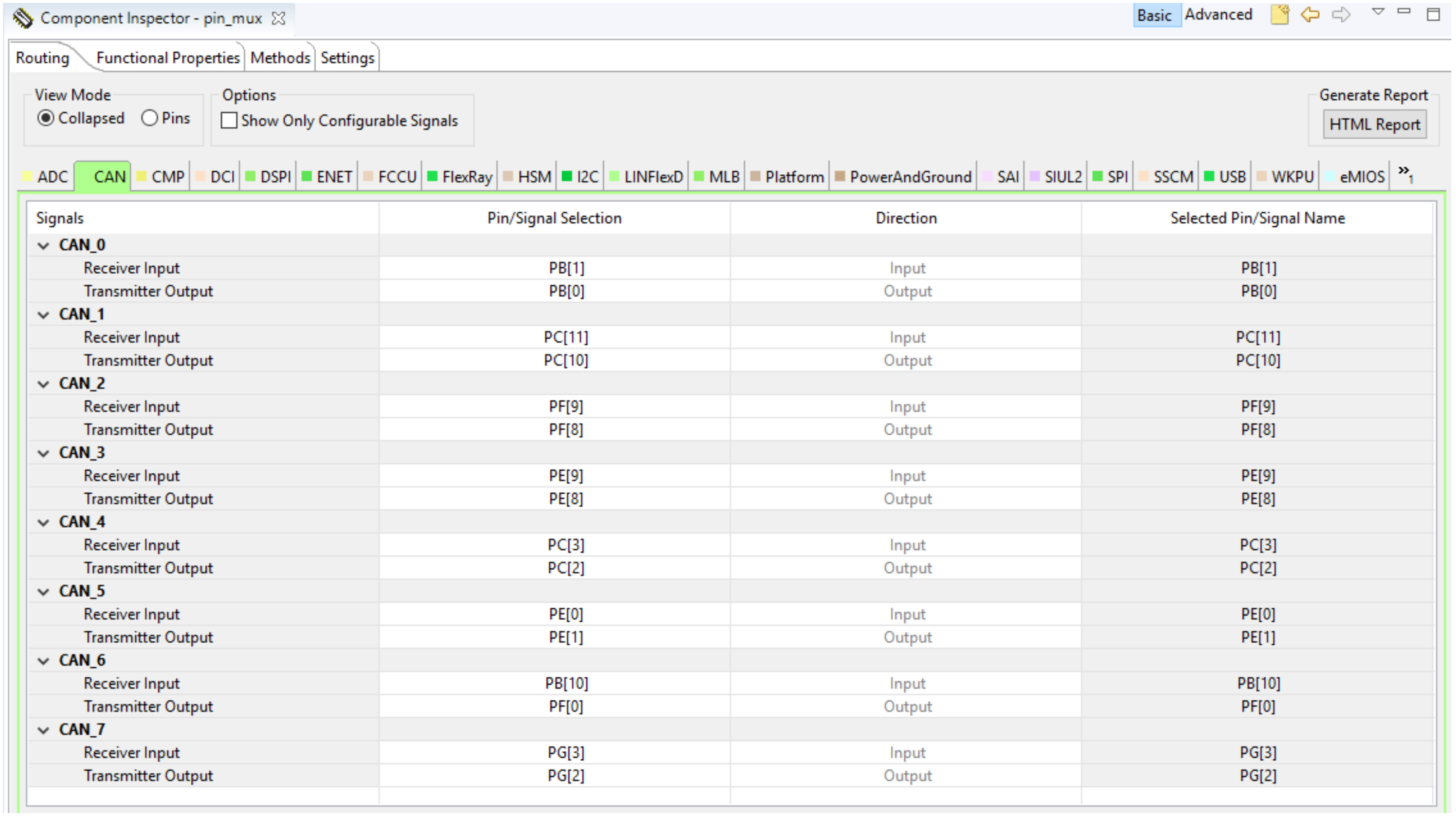
Pin	User Pin/Signal Name	SRC	OBE	ODE
5	PJ[4]	Half drive stre...	Disabled	Disabled
71	PJ[3]	Half drive stre...	Disabled	Disabled
72	PJ[2]	Half drive stre...	Disabled	Disabled
73	PJ[1]	Half drive stre...	Disabled	Disabled
74	PJ[0]	Half drive stre...	Disabled	Disabled

Note:The default configuration of PJ[4] is input, we need to change the OBE to Enabled mode enable output.

```
void CAN_TJA1043T_Enable(void)
{
    PINS_DRV_WritePin(PTB, 3, 1);      /* Initialize high CAN4 EN */
    PINS_DRV_WritePin(PTA, 2, 1);      /* Initialize high CAN5 EN */
    PINS_DRV_WritePin(PTG, 9, 1);      /* Initialize high CAN6 EN */
    PINS_DRV_WritePin(PTH, 14, 1);     /* Initialize high CAN7 EN */

    PINS_DRV_WritePin(PTJ, 4, 1);      /* Initialize high CAN4 STB */
    PINS_DRV_WritePin(PTE, 10, 1);     /* Initialize high CAN5 STB */
    PINS_DRV_WritePin(PTF, 12, 1);     /* Initialize high CAN6 STB */
    PINS_DRV_WritePin(PTI, 7, 1);      /* Initialize high CAN7 STB */
}
```

Hands-on – CAN: Modify- Ports of CAN

- In ‘pin_mux’ component –> Routing(Collapsed)– select ‘CAN’
  - Configuration the ports according to the schematic.


Component Inspector - pin_mux
Basic
Advanced
Routing
Functional Properties
Methods
Settings
Generate Report
HTML Report
ADC CAN CMP DCI DSPI ENET FCCU FlexRay HSM I2C LINFlexD MLB Platform PowerAndGround SAI SIUL2 SPI SSCM USB WKPU eMIOS

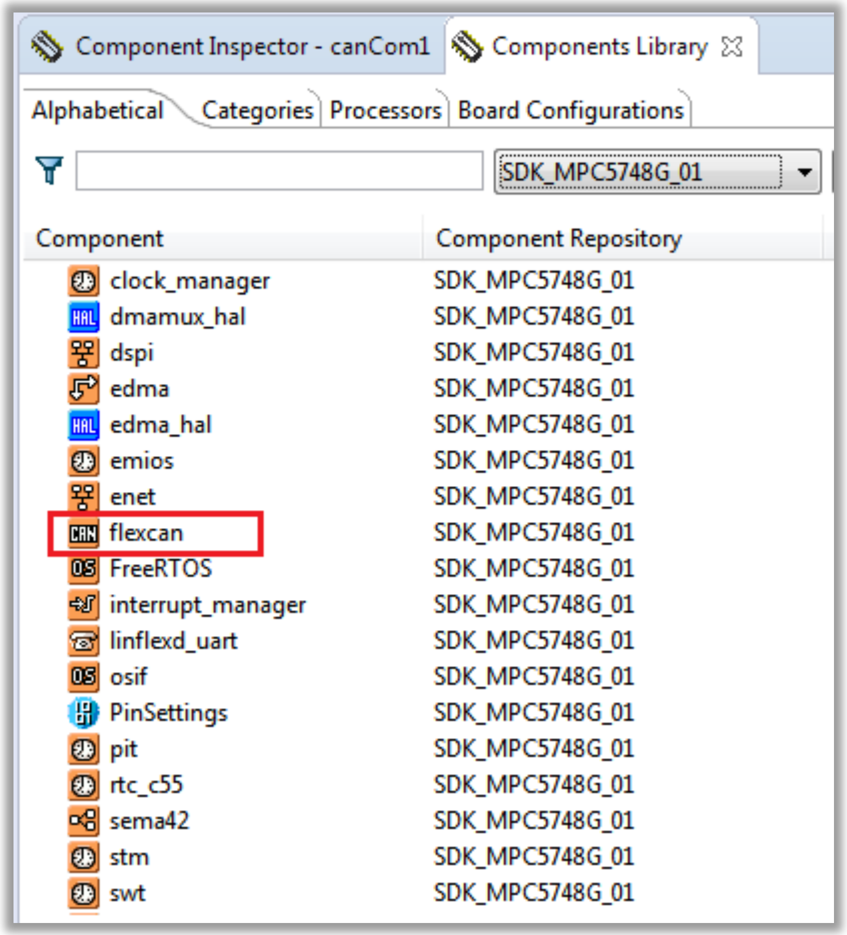
View Mode: [✓ Collapsed] Pins

Options: Show Only Configurable Signals

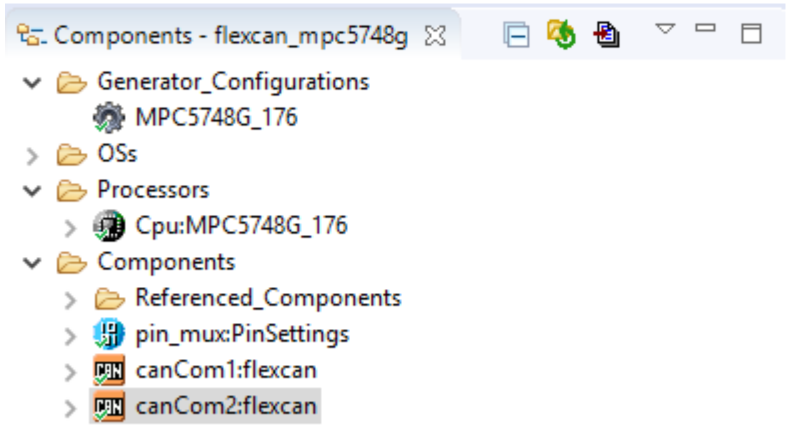
Signals	Pin/Signal Selection	Direction	Selected Pin/Signal Name
CAN_0			
Receiver Input	PB[1]	Input	PB[1]
Transmitter Output	PB[0]	Output	PB[0]
CAN_1			
Receiver Input	PC[11]	Input	PC[11]
Transmitter Output	PC[10]	Output	PC[10]
CAN_2			
Receiver Input	PF[9]	Input	PF[9]
Transmitter Output	PF[8]	Output	PF[8]
CAN_3			
Receiver Input	PE[9]	Input	PE[9]
Transmitter Output	PE[8]	Output	PE[8]
CAN_4			
Receiver Input	PC[3]	Input	PC[3]
Transmitter Output	PC[2]	Output	PC[2]
CAN_5			
Receiver Input	PE[0]	Input	PE[0]
Transmitter Output	PE[1]	Output	PE[1]
CAN_6			
Receiver Input	PB[10]	Input	PB[10]
Transmitter Output	PF[0]	Output	PF[0]
CAN_7			
Receiver Input	PG[3]	Input	PG[3]
Transmitter Output	PG[2]	Output	PG[2]

Hands-on – CAN: Modify- CAN configuration

- From ‘Components Library’ view, double-click ‘flexcan’ component to add it the project

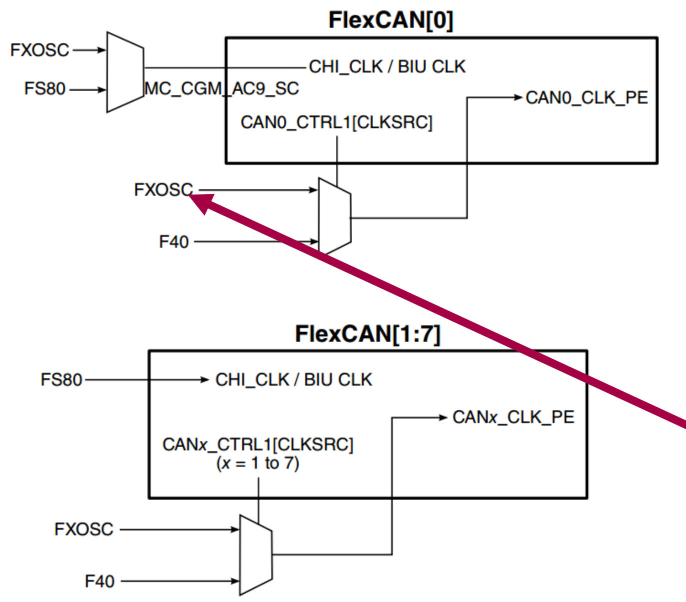


Component	Component Repository
clock_manager	SDK_MPC5748G_01
dmamux_hal	SDK_MPC5748G_01
dspi	SDK_MPC5748G_01
edma	SDK_MPC5748G_01
edma_hal	SDK_MPC5748G_01
emios	SDK_MPC5748G_01
enet	SDK_MPC5748G_01
flexcan	SDK_MPC5748G_01
FreeRTOS	SDK_MPC5748G_01
interrupt_manager	SDK_MPC5748G_01
linflexd_uart	SDK_MPC5748G_01
osif	SDK_MPC5748G_01
PinSettings	SDK_MPC5748G_01
pit	SDK_MPC5748G_01
rtc_c55	SDK_MPC5748G_01
sema42	SDK_MPC5748G_01
stm	SDK_MPC5748G_01
swt	SDK_MPC5748G_01

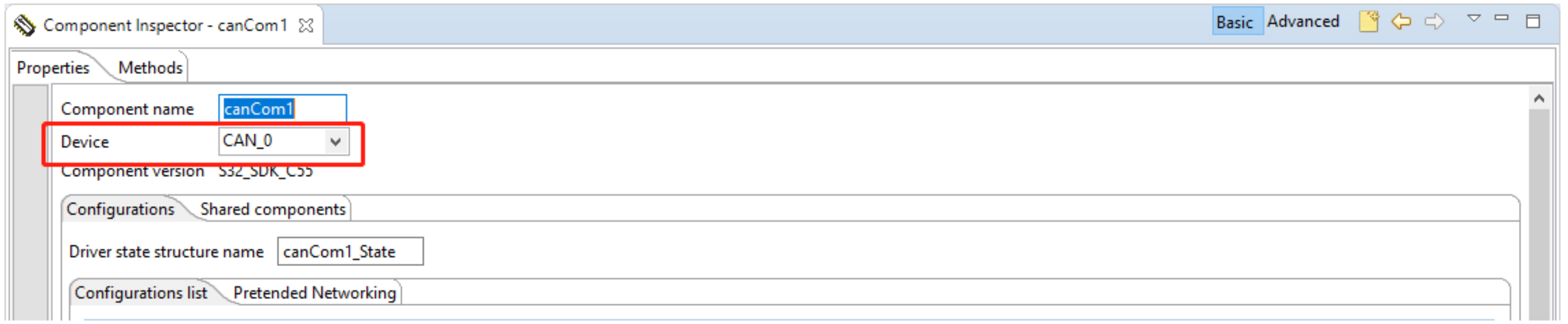


Hands-on – CAN: Modify- CAN configuration

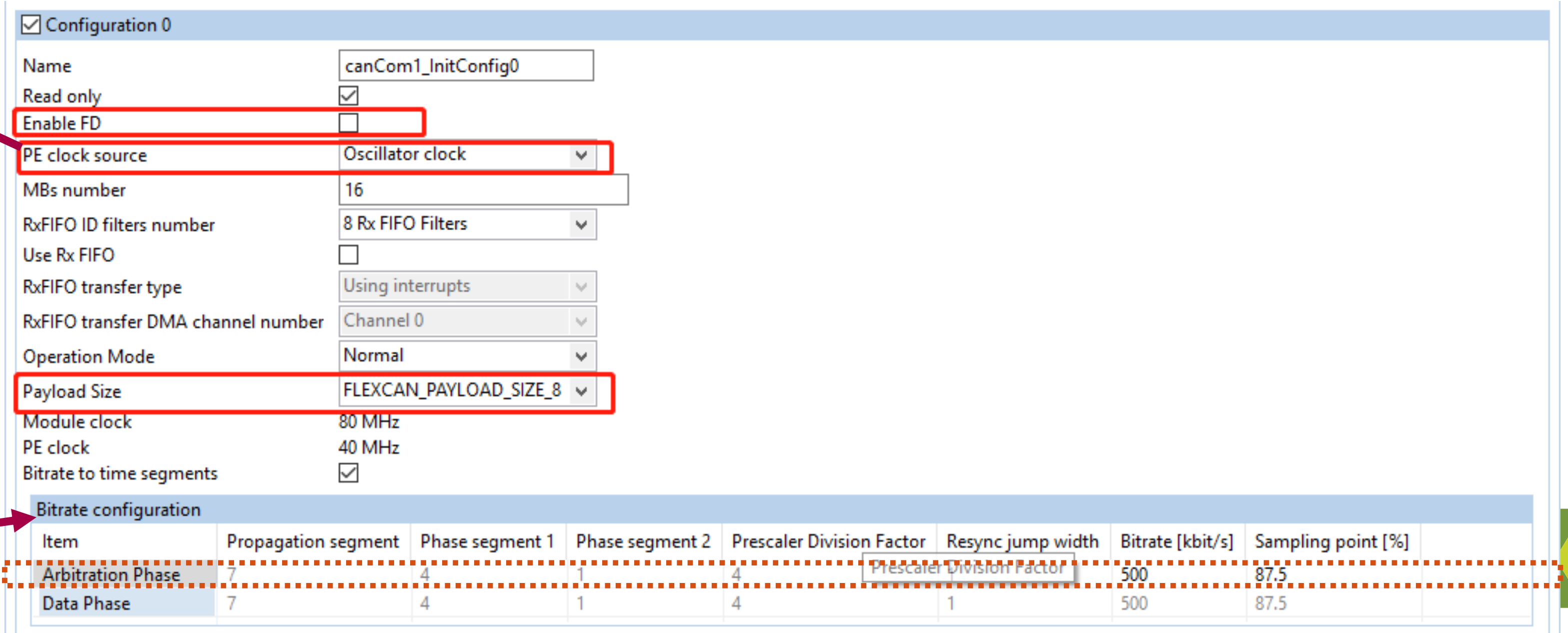
- No need to change default configuration for CAN:
  - standard CAN (no FD), minimum payload, 500 kbps



NOTE: The multiplexers for all FlexCAN modules are located outside of the module, but are controlled by registers inside the FlexCAN modules.



Component name	canCom1
Device	CAN_0
Component version	S32_SDK_C55
Driver state structure name	canCom1_State



Configuration 0

Name	canCom1_InitConfig0
Read only	✓
Enable FD	☐
PE clock source	Oscillator clock
MBs number	16
RxFIFO ID filters number	8 Rx FIFO Filters
Use Rx FIFO	☐
RxFIFO transfer type	Using interrupts
RxFIFO transfer DMA channel number	Channel 0
Operation Mode	Normal
Payload Size	FLEXCAN_PAYLOAD_SIZE_8
Module clock	80 MHz
PE clock	40 MHz
Bitrate to time segments	✓

Bitrate configuration

Item	Propagation segment	Phase segment 1	Phase segment 2	Prescaler Division Factor	Resync jump width	Bitrate [kbit/s]	Sampling point [%]
Arbitration Phase	7	4	1	4	[illegible]	500	87.5
Data Phase	7	4	1	4	1	500	87.5

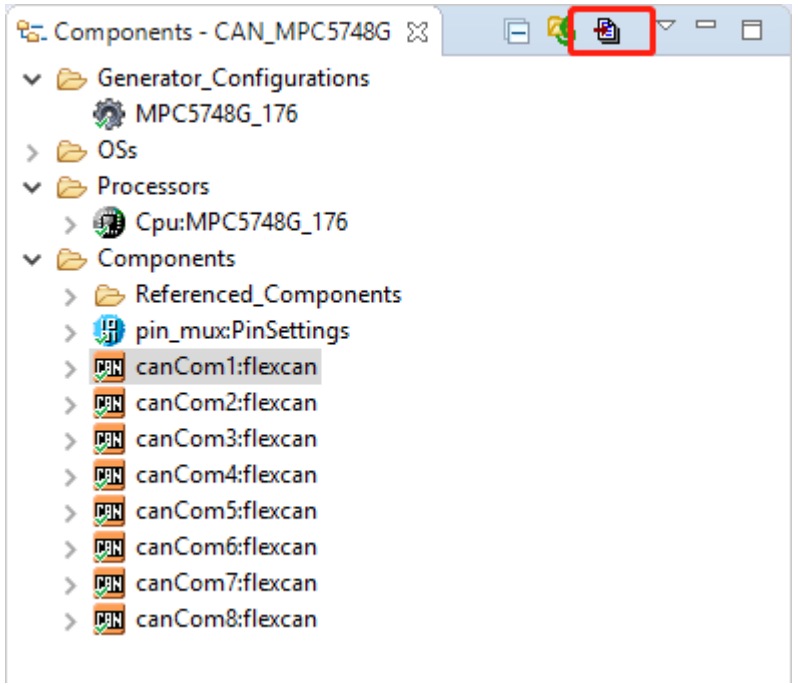
Equation:

PE clock(40MHz)=
((**propagation segment**+1)+(**segment1**+1)+(**segment2**+1)+1)
*(**Prescaler Division** +1)*(**Bitrate**)

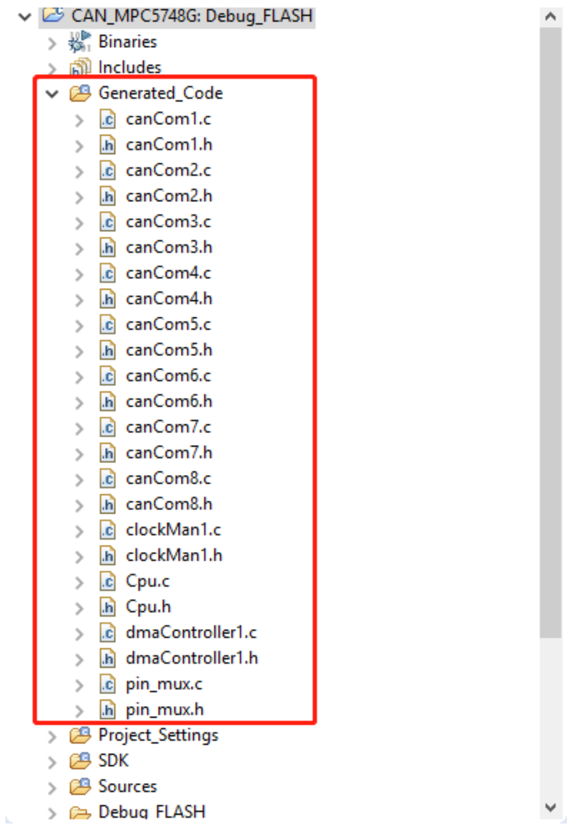
Hands-on – CAN: Modify- CAN configuration

- After all the CAN configurations are complete, the components window should be like this

-Hit the ‘Generate code’ button



```
Components - CAN_MPC5748G
v Generator_Configurations
    MPC5748G_176
> OSs
v Processors
  > Cpu:MPC5748G_176
v Components
  > Referenced_Components
  > pin_mux:PinSettings
  > canCom1:flexcan
  > canCom2:flexcan
  > canCom3:flexcan
  > canCom4:flexcan
  > canCom5:flexcan
  > canCom6:flexcan
  > canCom7:flexcan
  > canCom8:flexcan
```



```
v CAN_MPC5748G: Debug_FLASH
  > Binaries
  > Includes
  v Generated_Code
    > canCom1.c
    > canCom1.h
    > canCom2.c
    > canCom2.h
    > canCom3.c
    > canCom3.h
    > canCom4.c
    > canCom4.h
    > canCom5.c
    > canCom5.h
    > canCom6.c
    > canCom6.h
    > canCom7.c
    > canCom7.h
    > canCom8.c
    > canCom8.h
    > clockMan1.c
    > clockMan1.h
    > Cpu.c
    > Cpu.h
    > dmaController1.c
    > dmaController1.h
    > pin_mux.c
    > pin_mux.h
  > Project_Settings
  > SDK
  > Sources
  > Debug_FLASH
```

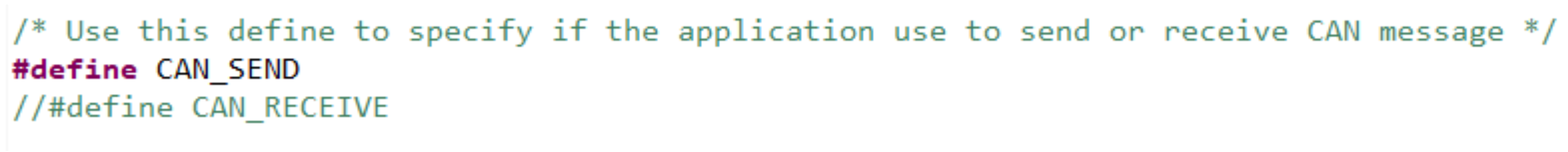
Hands-on – CAN: Application Code

- Open the main.c file in text editor view

  You can refer to the sample project (**can_mpc5748g**) we provided to modify the main.c file. You can also replace it directly.

Note:

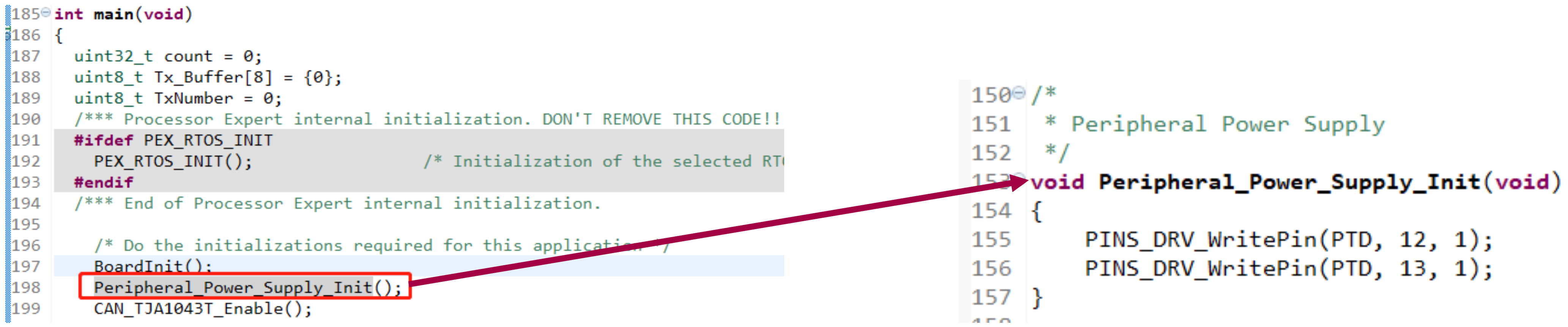
The sending and receiving function of this project is realized separately, which needs to be controlled by macros.



```
/* Use this define to specify if the application use to send or receive CAN message */
#define CAN_SEND
//#define CAN_RECEIVE
```

Hands-on – CAN: Application Code

(1) Peripheral Power Supply:

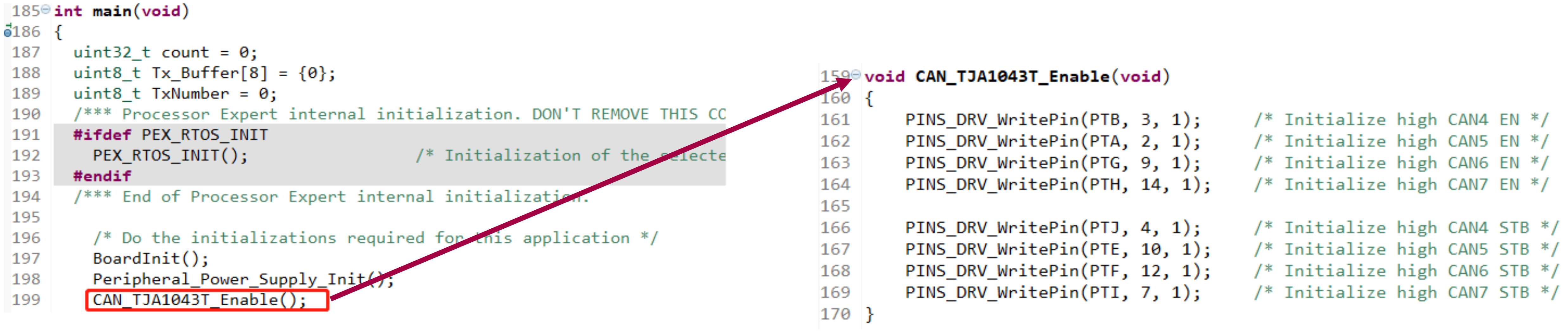


```
int main(void)
{
  uint32_t count = 0;
  uint8_t Tx_Buffer[8] = {0};
  uint8_t TxNumber = 0;
  /*** Processor Expert internal initialization. DON'T REMOVE THIS CODE!!
  #ifdef PEX_RTOS_INIT
    PEX_RTOS_INIT();                 /* Initialization of the selected RT
  #endif
  /*** End of Processor Expert internal initialization.

    /* Do the initializations required for this application */
    BoardInit();
    Peripheral_Power_Supply_Init();
    CAN_TJA1043T_Enable();
```

```
/*
 * Peripheral Power Supply
 */
void Peripheral_Power_Supply_Init(void)
{
    PINS_DRV_WritePin(PTD, 12, 1);
    PINS_DRV_WritePin(PTD, 13, 1);
}
```

Hands-on – CAN: Application Code

(2) CAN4~CAN7 `S Phy（TJA1043T）Enable:



```
int main(void)
{
  uint32_t count = 0;
  uint8_t Tx_Buffer[8] = {0};
  uint8_t TxNumber = 0;
  /*** Processor Expert internal initialization. DON'T REMOVE THIS CC
  #ifdef PEX_RTOS_INIT
    PEX_RTOS_INIT();                 /* Initialization of the selecte
  #endif
  /*** End of Processor Expert internal initialization.

    /* Do the initializations required for this application */
    BoardInit();
    Peripheral_Power_Supply_Init();
    CAN_TJA1043T_Enable();
```

```
void CAN_TJA1043T_Enable(void)
{
    PINS_DRV_WritePin(PTB, 3, 1);      /* Initialize high CAN4 EN */
    PINS_DRV_WritePin(PTA, 2, 1);      /* Initialize high CAN5 EN */
    PINS_DRV_WritePin(PTG, 9, 1);      /* Initialize high CAN6 EN */
    PINS_DRV_WritePin(PTH, 14, 1);     /* Initialize high CAN7 EN */

    PINS_DRV_WritePin(PTJ, 4, 1);      /* Initialize high CAN4 STB */
    PINS_DRV_WritePin(PTE, 10, 1);     /* Initialize high CAN5 STB */
    PINS_DRV_WritePin(PTF, 12, 1);     /* Initialize high CAN6 STB */
    PINS_DRV_WritePin(PTI, 7, 1);      /* Initialize high CAN7 STB */
}
```

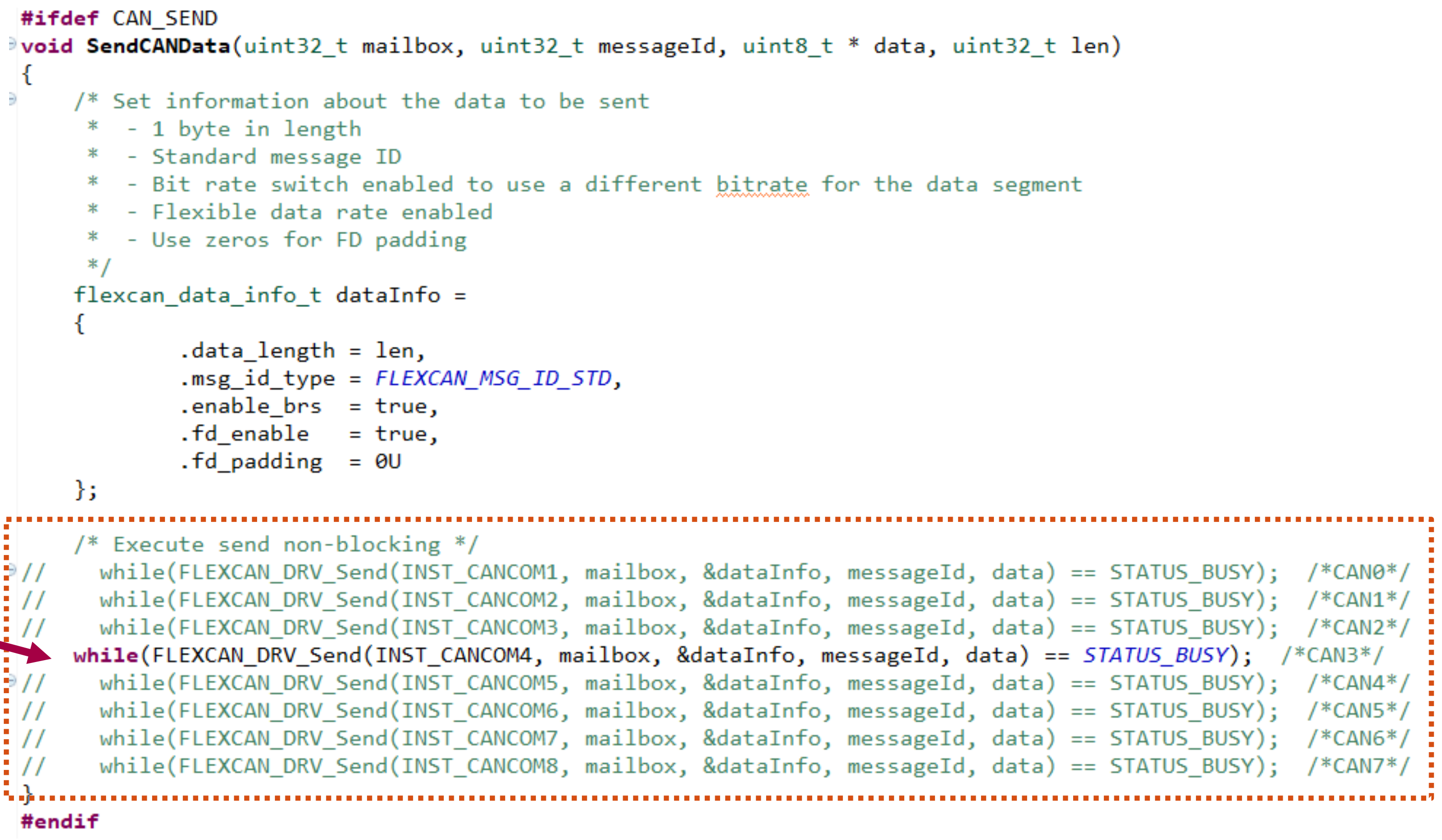
Hands-on – CAN: Application Code

(3) Sending data via CAN:

Note:

When testing , we need to connect the port of CAN which need tested, otherwise it will wait for the completion of sending.

Eg. Testing CAN3



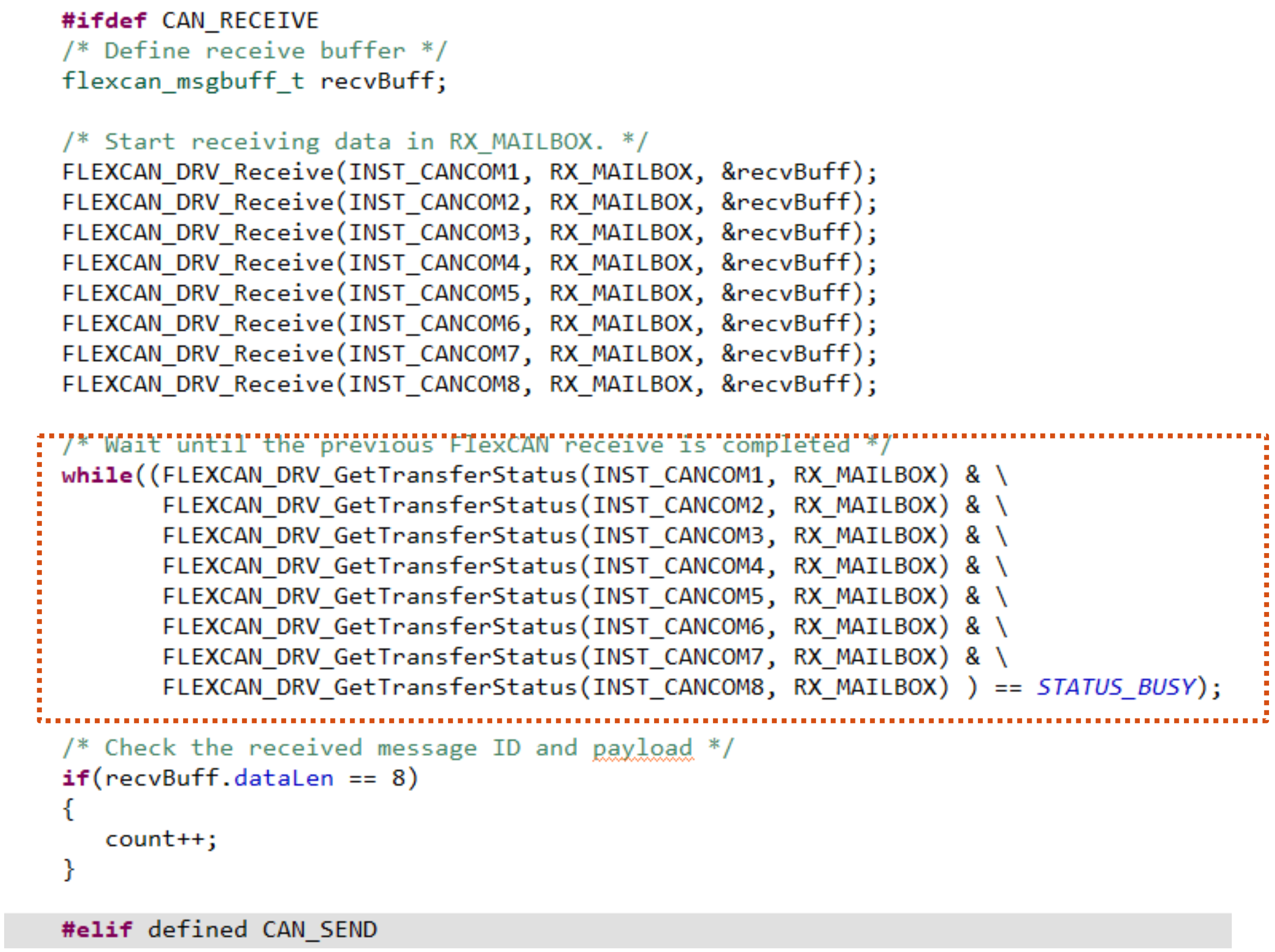
```
#ifdef CAN_SEND
void SendCANData(uint32_t mailbox, uint32_t messageId, uint8_t * data, uint32_t len)
{
    /* Set information about the data to be sent
     *  - 1 byte in length
     *  - Standard message ID
     *  - Bit rate switch enabled to use a different bitrate for the data segment
     *  - Flexible data rate enabled
     *  - Use zeros for FD padding
     */
    flexcan_data_info_t dataInfo =
    {
            .data_length = len,
            .msg_id_type = FLEXCAN_MSG_ID_STD,
            .enable_brs  = true,
            .fd_enable   = true,
            .fd_padding  = 0U
    };

    /* Execute send non-blocking */
//    while(FLEXCAN_DRV_Send(INST_CANCOM1, mailbox, &dataInfo, messageId, data) == STATUS_BUSY);  /*CAN0*/
//    while(FLEXCAN_DRV_Send(INST_CANCOM2, mailbox, &dataInfo, messageId, data) == STATUS_BUSY);  /*CAN1*/
//    while(FLEXCAN_DRV_Send(INST_CANCOM3, mailbox, &dataInfo, messageId, data) == STATUS_BUSY);  /*CAN2*/
    while(FLEXCAN_DRV_Send(INST_CANCOM4, mailbox, &dataInfo, messageId, data) == STATUS_BUSY);  /*CAN3*/
//    while(FLEXCAN_DRV_Send(INST_CANCOM5, mailbox, &dataInfo, messageId, data) == STATUS_BUSY);  /*CAN4*/
//    while(FLEXCAN_DRV_Send(INST_CANCOM6, mailbox, &dataInfo, messageId, data) == STATUS_BUSY);  /*CAN5*/
//    while(FLEXCAN_DRV_Send(INST_CANCOM7, mailbox, &dataInfo, messageId, data) == STATUS_BUSY);  /*CAN6*/
//    while(FLEXCAN_DRV_Send(INST_CANCOM8, mailbox, &dataInfo, messageId, data) == STATUS_BUSY);  /*CAN7*/
}
#endif
```

Hands-on – CAN: Application Code

(4) Receiving data via CAN:

Note:

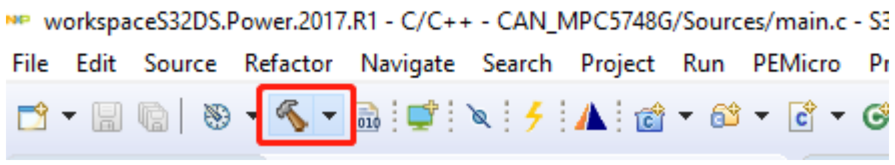
Unlike CAN sending, you can test the receive on any port.

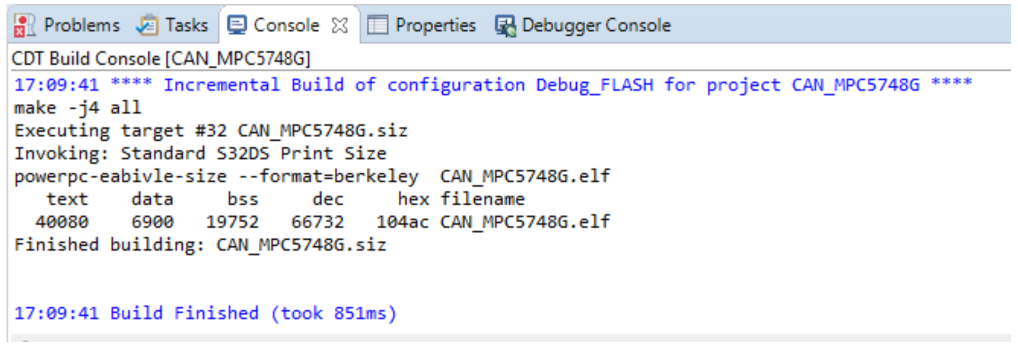


```
#ifdef CAN_RECEIVE
/* Define receive buffer */
flexcan_msgbuff_t recvBuff;

/* Start receiving data in RX_MAILBOX. */
FLEXCAN_DRV_Receive(INST_CANCOM1, RX_MAILBOX, &recvBuff);
FLEXCAN_DRV_Receive(INST_CANCOM2, RX_MAILBOX, &recvBuff);
FLEXCAN_DRV_Receive(INST_CANCOM3, RX_MAILBOX, &recvBuff);
FLEXCAN_DRV_Receive(INST_CANCOM4, RX_MAILBOX, &recvBuff);
FLEXCAN_DRV_Receive(INST_CANCOM5, RX_MAILBOX, &recvBuff);
FLEXCAN_DRV_Receive(INST_CANCOM6, RX_MAILBOX, &recvBuff);
FLEXCAN_DRV_Receive(INST_CANCOM7, RX_MAILBOX, &recvBuff);
FLEXCAN_DRV_Receive(INST_CANCOM8, RX_MAILBOX, &recvBuff);

/* Wait until the previous FlexCAN receive is completed */
while((FLEXCAN_DRV_GetTransferStatus(INST_CANCOM1, RX_MAILBOX) & \
       FLEXCAN_DRV_GetTransferStatus(INST_CANCOM2, RX_MAILBOX) & \
       FLEXCAN_DRV_GetTransferStatus(INST_CANCOM3, RX_MAILBOX) & \
       FLEXCAN_DRV_GetTransferStatus(INST_CANCOM4, RX_MAILBOX) & \
       FLEXCAN_DRV_GetTransferStatus(INST_CANCOM5, RX_MAILBOX) & \
       FLEXCAN_DRV_GetTransferStatus(INST_CANCOM6, RX_MAILBOX) & \
       FLEXCAN_DRV_GetTransferStatus(INST_CANCOM7, RX_MAILBOX) & \
       FLEXCAN_DRV_GetTransferStatus(INST_CANCOM8, RX_MAILBOX) ) == STATUS_BUSY);
/* Check the received message ID and payload */
if(recvBuff.dataLen == 8)
{
   count++;
}

#elif defined CAN_SEND
```

Hands-on – CAN: Build and Debug

• Click the ‘build project’ button – make sure there are no compilation errors

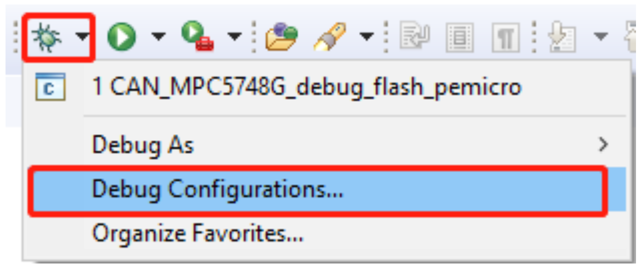

workspaceS32DS.Power.2017.R1 - C/C++ - CAN_MPC5748G/Sources/main.c - S3
File Edit Source Refactor Navigate Search Project Run PEMicro Pr

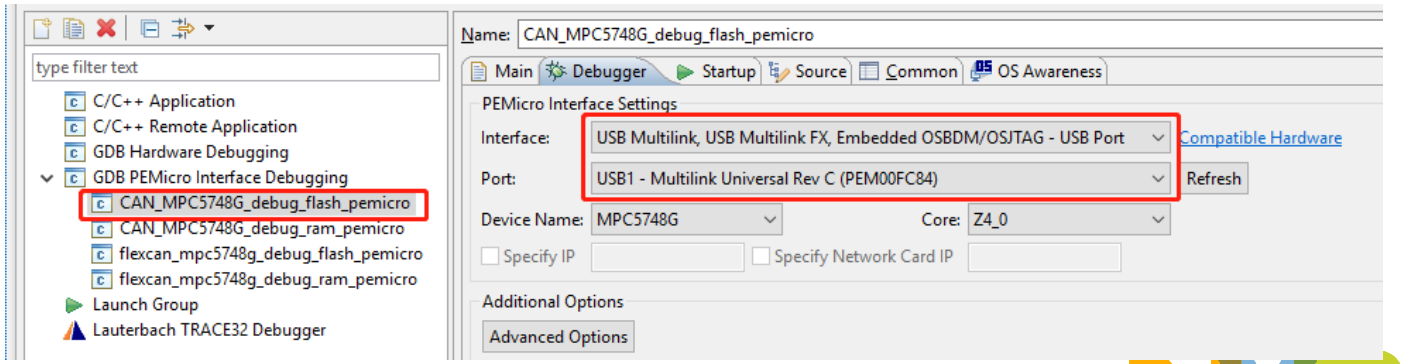


CDT Build Console [CAN_MPC5748G]

```
17:09:41 **** Incremental Build of configuration Debug_FLASH for project CAN_MPC5748G ****
make -j4 all
Executing target #32 CAN_MPC5748G.siz
Invoking: Standard S32DS Print Size
powerpc-eabivle-size --format=berkeley  CAN_MPC5748G.elf
   text    data     bss     dec     hex filename
  40080    6900   19752   66732   104ac CAN_MPC5748G.elf
Finished building: CAN_MPC5748G.siz


17:09:41 Build Finished (took 851ms)
```

• Select the correct debug configuration and interface to debug the application




type filter text
C/C++ Application
C/C++ Remote Application
GDB Hardware Debugging
GDB PEMicro Interface Debugging
CAN_MPC5748G_debug_flash_pemicro
CAN_MPC5748G_debug_ram_pemicro
flexcan_mpc5748g_debug_flash_pemicro
flexcan_mpc5748g_debug_ram_pemicro
Launch Group
Lauterbach TRACE32 Debugger
Main Debugger Startup Source Common OS Awareness
Compatible Hardware
Refresh
Additional Options
Advanced Options

Name: CAN_MPC5748G_debug_flash_pemicro

PEMicro Interface Settings

Interface: USB Multilink, USB Multilink FX, Embedded OSBDM/OSJTAG - USB Port

Port: USB1 - Multilink Universal Rev C (PEM00FC84)

Device Name: MPC5748G Core: Z4_0

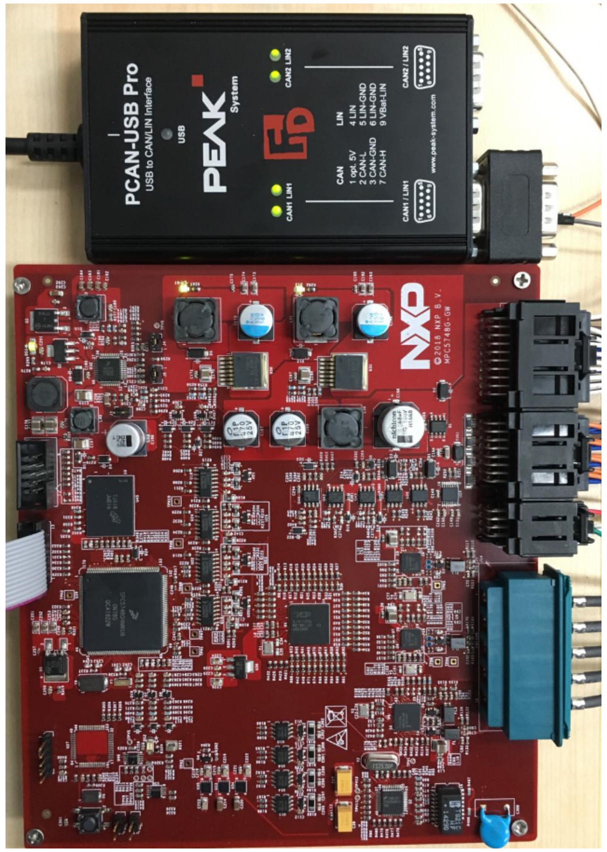
☐ Specify IP ☐ Specify Network Card IP

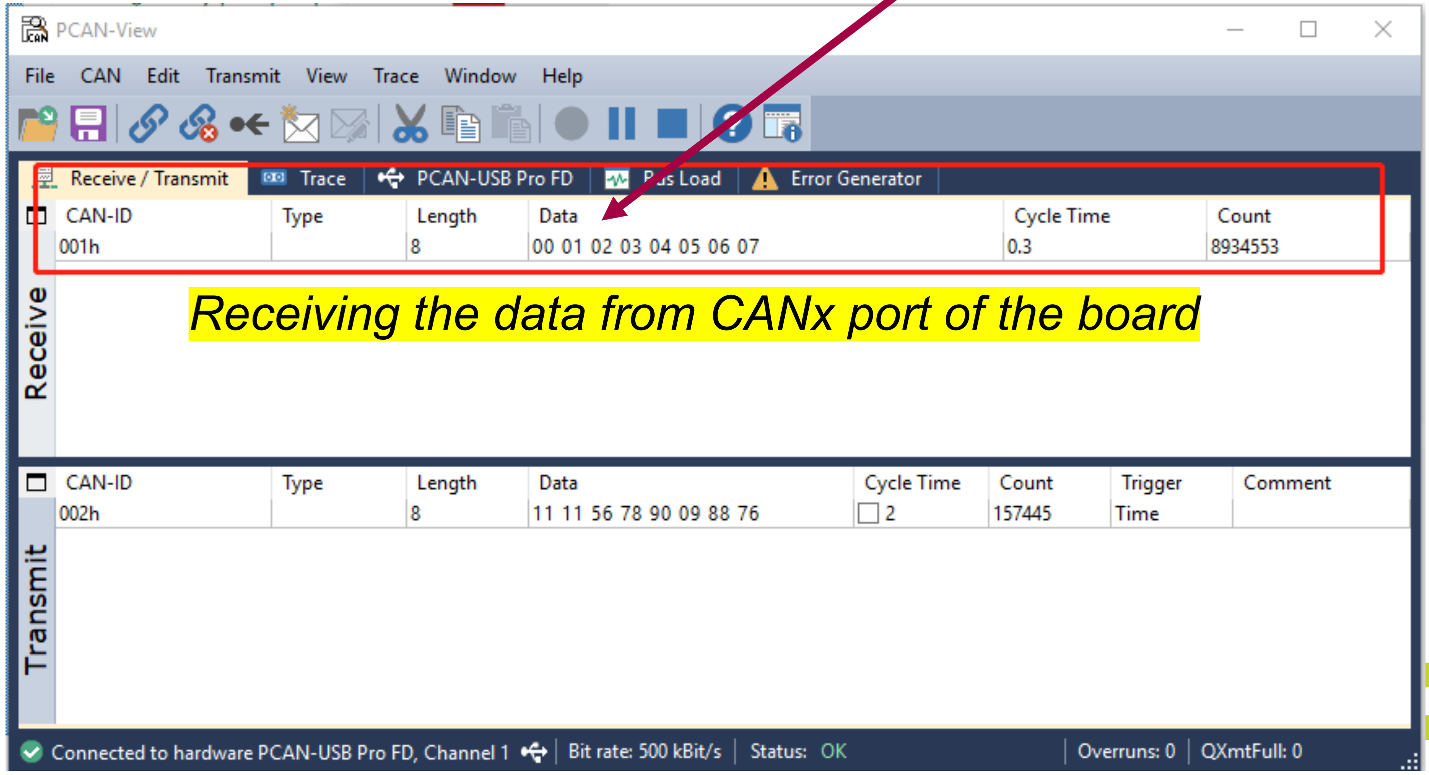
Hands-on – CAN: Build and Debug

```
#elif defined CAN_SEND
for(TxNumber=0; TxNumber<8; TxNumber++)
{
    Tx_Buffer[TxNumber] = TxNumber;
}
```

The data of CAN to be send

- **TEST: Sending data via CAN:**
  1) Open the macro of CAN_SEND, commented out CAN_RECEIVE.
  2) Connect the sending port with PEAK(The test tools we use) through the wire.
  3) Debug the application.





Receive:

CAN-ID	Type	Length	Data	Cycle Time	Count
001h		8	00 01 02 03 04 05 06 07	0.3	8934553

Receiving the data from CANx port of the board

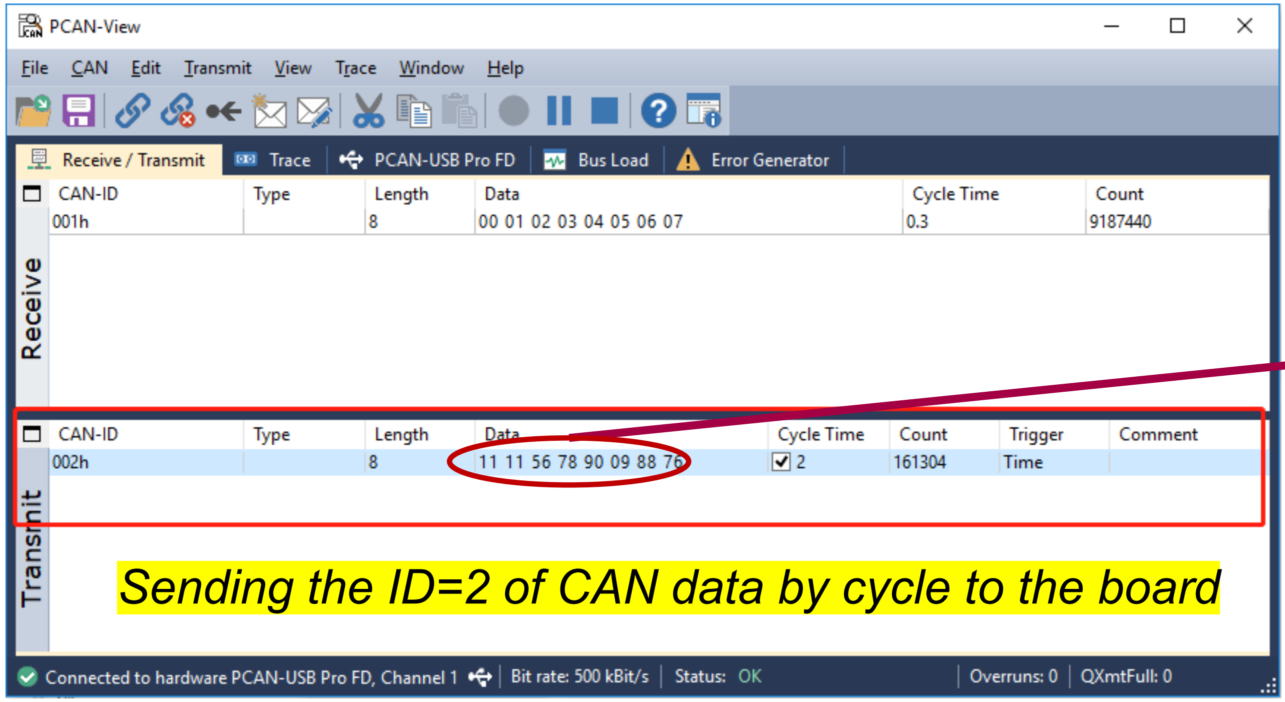
Transmit:

CAN-ID	Type	Length	Data	Cycle Time	Count	Trigger	Comment
002h		8	11 11 56 78 90 09 88 76	☐ 2	157445	Time	

Connected to hardware PCAN-USB Pro FD, Channel 1 | Bit rate: 500 kBit/s | Status: OK | Overruns: 0 | QXmtFull: 0

Hands-on – CAN: Build and Debug

- **TEST: Receiving data via CAN :**
  1) Open the macro of CAN_RECEIVE, commented out CAN_SEND
  2) Connect the receiving port with PEAK(The test tools we use) through the wire.
  3) Debug the application.



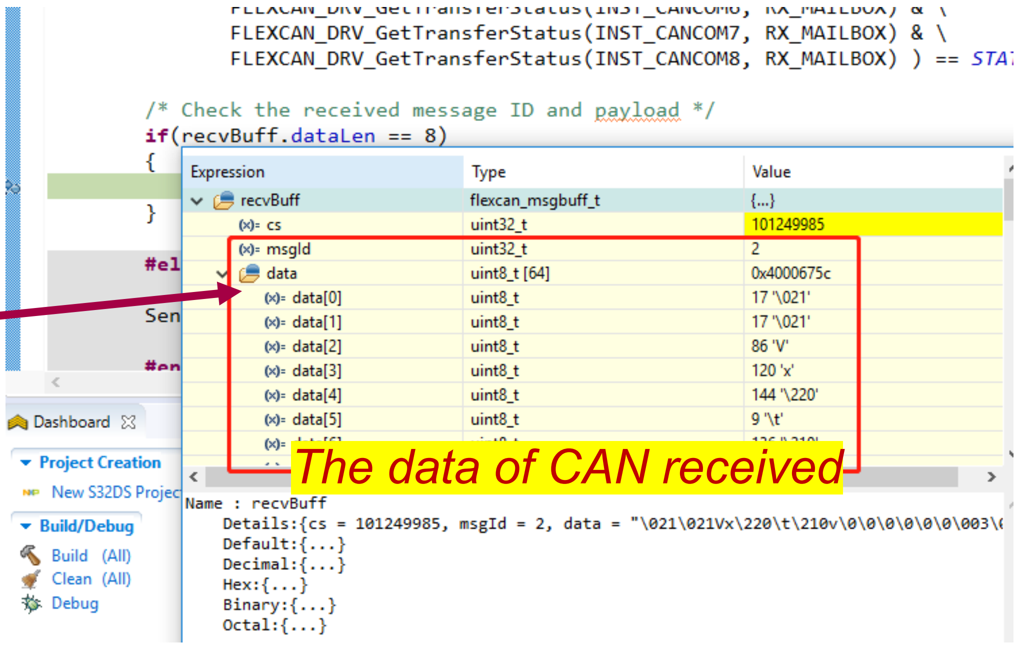
CAN-ID	Type	Length	Data	Cycle Time	Count
001h		8	00 01 02 03 04 05 06 07	0.3	9187440

CAN-ID	Type	Length	Data	Cycle Time	Count	Trigger	Comment
002h		8	11 11 56 78 90 09 88 76	✓ 2	161304	Time	

Connected to hardware PCAN-USB Pro FD, Channel 1 | Bit rate: 500 kBit/s | Status: OK | Overruns: 0 | QXmtFull: 0

Sending the ID=2 of CAN data by cycle to the board

Debug window



```
FLEXCAN_DRV_GetTransferStatus(INST_CANCOM7, RX_MAILBOX) & \
FLEXCAN_DRV_GetTransferStatus(INST_CANCOM8, RX_MAILBOX) ) == STA

/* Check the received message ID and payload */
if(recvBuff.dataLen == 8)
{

}

#el

Sen

#en
```

Expression	Type	Value
recvBuff	flexcan_msgbuff_t	{...}
cs	uint32_t	101249985
msgId	uint32_t	2
data	uint8_t [64]	0x4000675c
data[0]	uint8_t	17 '\021'
data[1]	uint8_t	17 '\021'
data[2]	uint8_t	86 'V'
data[3]	uint8_t	120 'x'
data[4]	uint8_t	144 '\220'
data[5]	uint8_t	9 '\t'

```
Name : recvBuff
    Details:{cs = 101249985, msgId = 2, data = "\021\021Vx\220\t\210v\0\0\0\0\0\0\003\
    Default:{...}
    Decimal:{...}
    Hex:{...}
    Binary:{...}
    Octal:{...}
```

The data of CAN received

02.

Hands-on – CAN_FD

Hands-on – CAN_FD: Objective

- CAN_FD is configured similarly to CAN.
- This section will be modified based on the previous section(**Hands-on-CAN**).

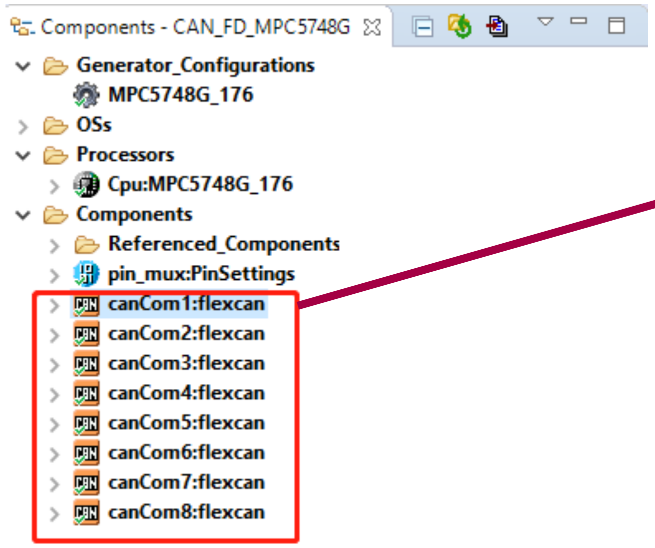
  - Modify the CAN configurations.

  - Modify the application code of main.c

- Please refer to section 1 if there is anything unclear

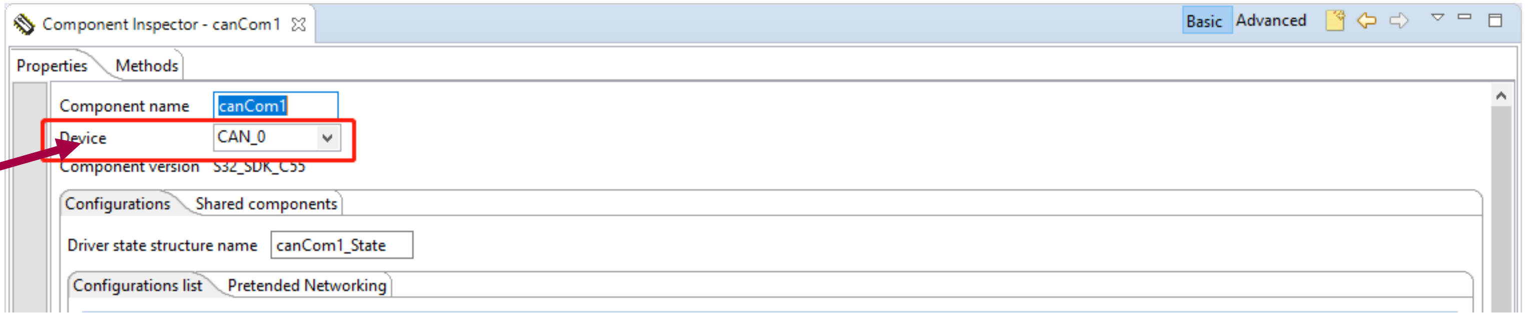
Hands-on – CAN_FD: Modify- CAN_FD configuration

- Need to change default configuration for CAN_FD:
  - standard CAN_FD, maxmum payload, Arbitration Phase: 500 kbps, Data Phase: 2000kbps.



Components - CAN_FD_MPC5748G

- Generator_Configurations
  - MPC5748G_176
- OSs
- Processors
  - Cpu:MPC5748G_176
- Components
  - Referenced_Components
  - pin_mux:PinSettings
  - canCom1:flexcan
  - canCom2:flexcan
  - canCom3:flexcan
  - canCom4:flexcan
  - canCom5:flexcan
  - canCom6:flexcan
  - canCom7:flexcan
  - canCom8:flexcan



Component Inspector - canCom1

Basic | Advanced

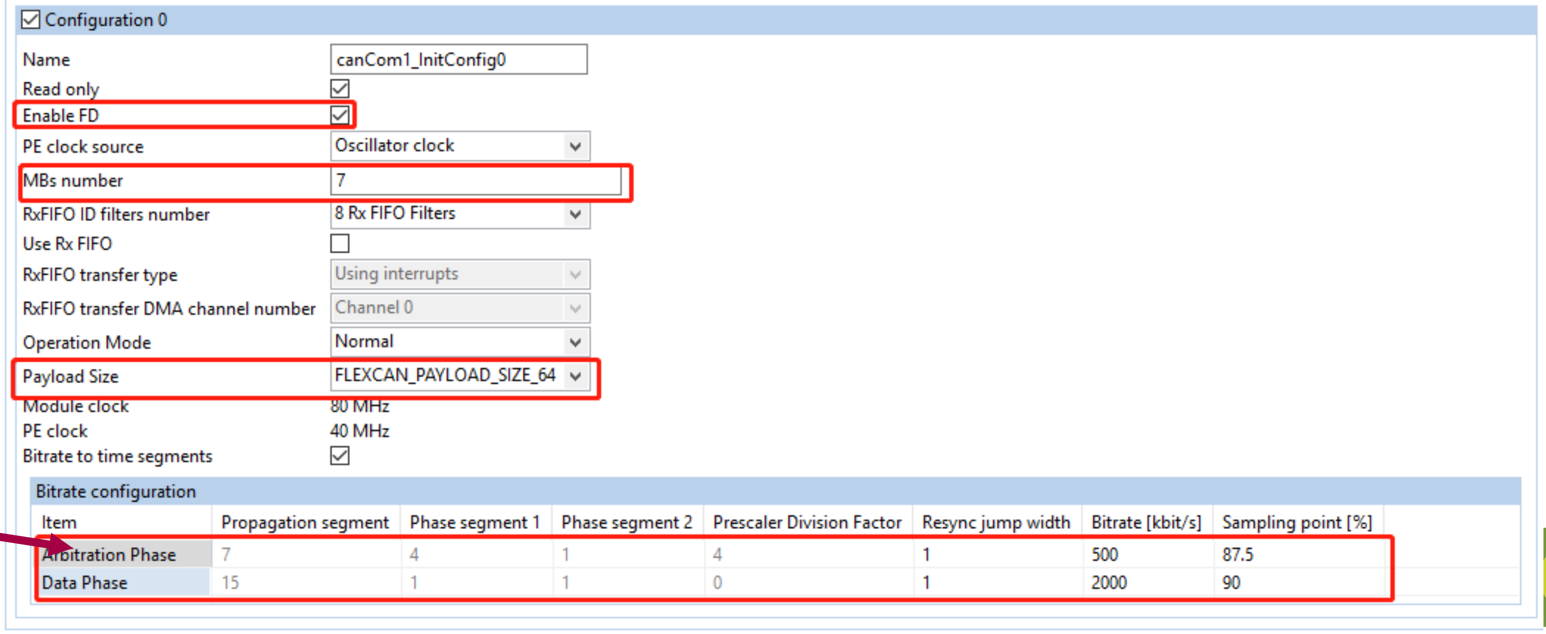
Properties | Methods

Component name	canCom1
Device	CAN_0
Component version	S32_SDK_C55

Configurations | Shared components

Driver state structure name	canCom1_State

Configurations list | Pretended Networking



Configuration 0

Name	canCom1_InitConfig0
Read only	✓
Enable FD	✓
PE clock source	Oscillator clock
MBs number	7
RxFIFO ID filters number	8 Rx FIFO Filters
Use Rx FIFO	☐
RxFIFO transfer type	Using interrupts
RxFIFO transfer DMA channel number	Channel 0
Operation Mode	Normal
Payload Size	FLEXCAN_PAYLOAD_SIZE_64
Module clock	80 MHz
PE clock	40 MHz
Bitrate to time segments	✓

Bitrate configuration

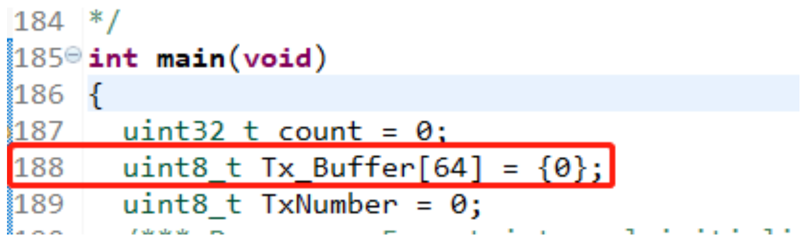
Item	Propagation segment	Phase segment 1	Phase segment 2	Prescaler Division Factor	Resync jump width	Bitrate [kbit/s]	Sampling point [%]
Arbitration Phase	7	4	1	4	1	500	87.5
Data Phase	15	1	1	0	1	2000	90

Equation:

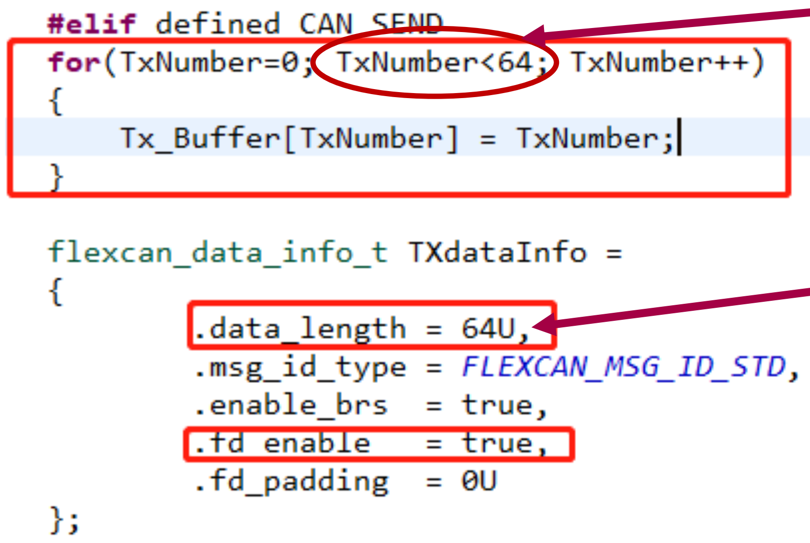
PE clock(40MHz)=
((propagation segment+1)+(segment1+1)+(segment2+1)+1)
(Prescaler Division +1)(Bitrate)

Hands-on – CAN_FD: Modify - Application Code

(1) Sending data via CAN:



```
*/
int main(void)
{
  uint32_t count = 0;
  uint8_t Tx_Buffer[64] = {0};
  uint8_t TxNumber = 0;
```



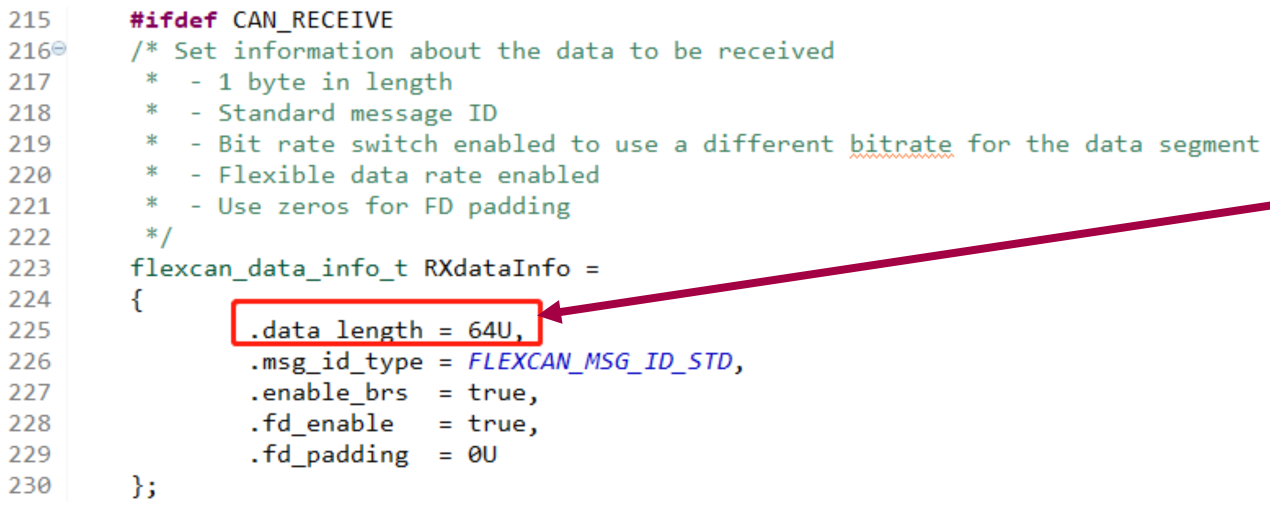
```
#elif defined CAN_SEND
for(TxNumber=0; TxNumber<64; TxNumber++)
{
    Tx_Buffer[TxNumber] = TxNumber;
}

flexcan_data_info_t TXdataInfo =
{
        .data_length = 64U,
        .msg_id_type = FLEXCAN_MSG_ID_STD,
        .enable_brs  = true,
        .fd_enable   = true,
        .fd_padding  = 0U
};
```

64Bytes of data to be send
(Data: from 0 to 63)

The length of the data to be send, the max length is 64

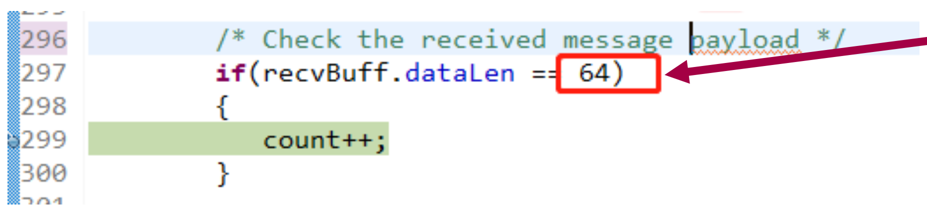
Hands-on – CAN_FD: Modify - Application Code

(2) Receiving data via CAN:



```
#ifdef CAN_RECEIVE
/* Set information about the data to be received
 *   - 1 byte in length
 *   - Standard message ID
 *   - Bit rate switch enabled to use a different bitrate for the data segment
 *   - Flexible data rate enabled
 *   - Use zeros for FD padding
 */
flexcan_data_info_t RXdataInfo =
{
        .data_length = 64U,
        .msg_id_type = FLEXCAN_MSG_ID_STD,
        .enable_brs  = true,
        .fd_enable   = true,
        .fd_padding  = 0U
};
```

The length of the data to be receive, the max length is 64



```
        /* Check the received message payload */
        if(recvBuff.dataLen == 64)
        {
            count++;
        }
```

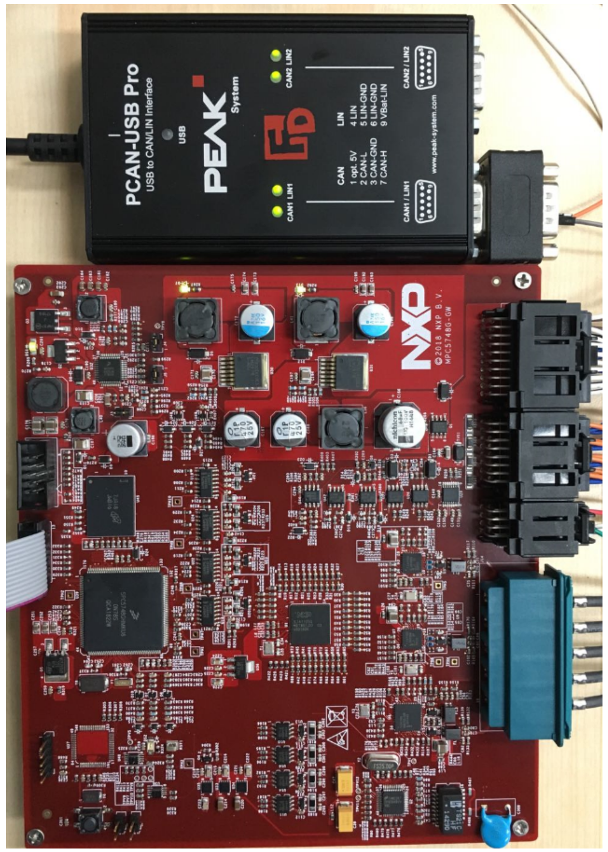
For debug

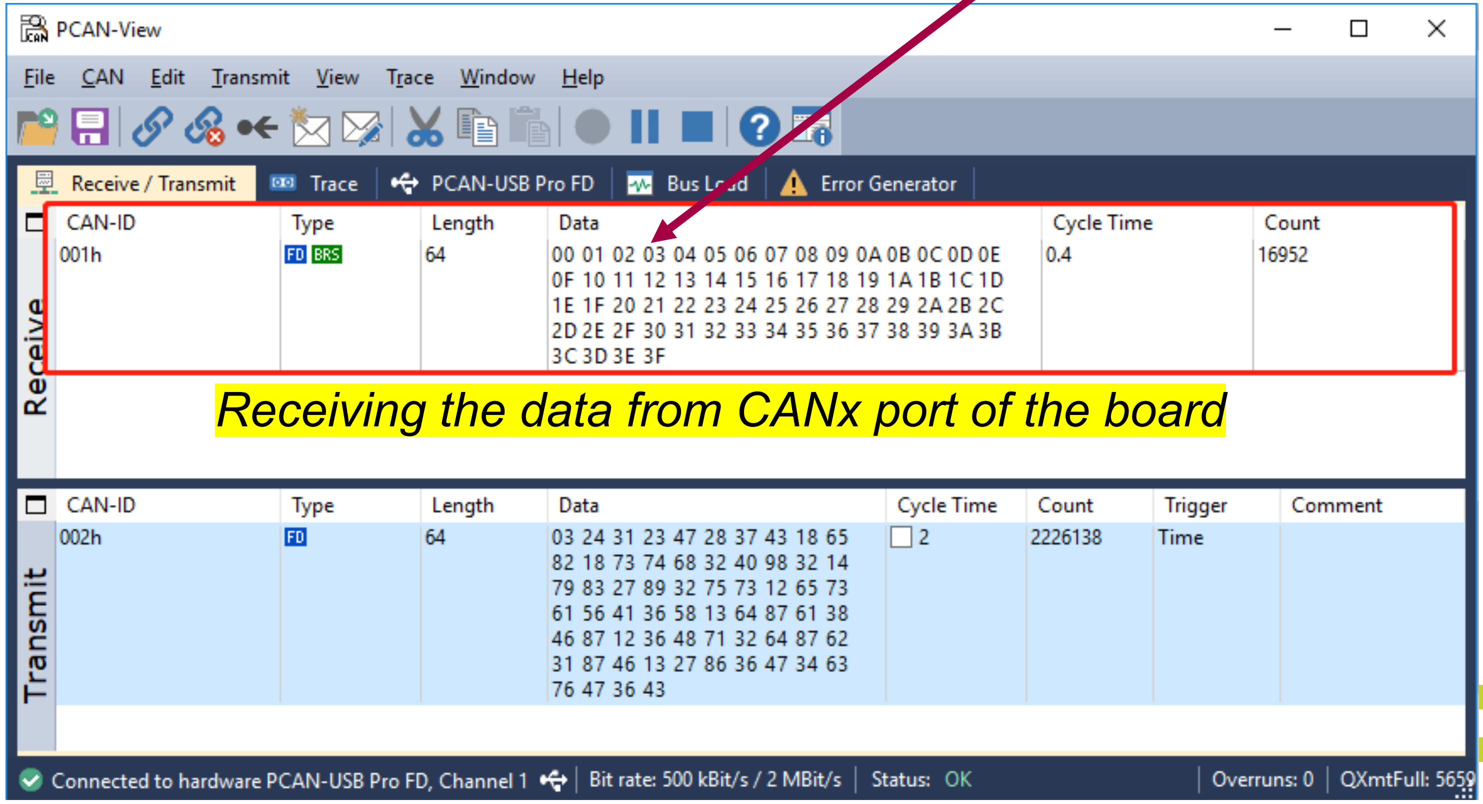
Hands-on – CAN_FD: Build and Debug

```
for(TxNumber=0; TxNumber<64; TxNumber++)
{
    Tx_Buffer[TxNumber] = TxNumber;
}
```

The data of CAN to be send

- **TEST: Sending data via CAN_FD:**
  1) Open the macro of CAN_SEND, commented out CAN_RECEIVE.
  2) Connect the sending port with PEAK(The test tools we use) through the wire.
  3) Debug the application.





CAN-ID	Type	Length	Data	Cycle Time	Count
001h	FD BRS	64	00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F 30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F	0.4	16952

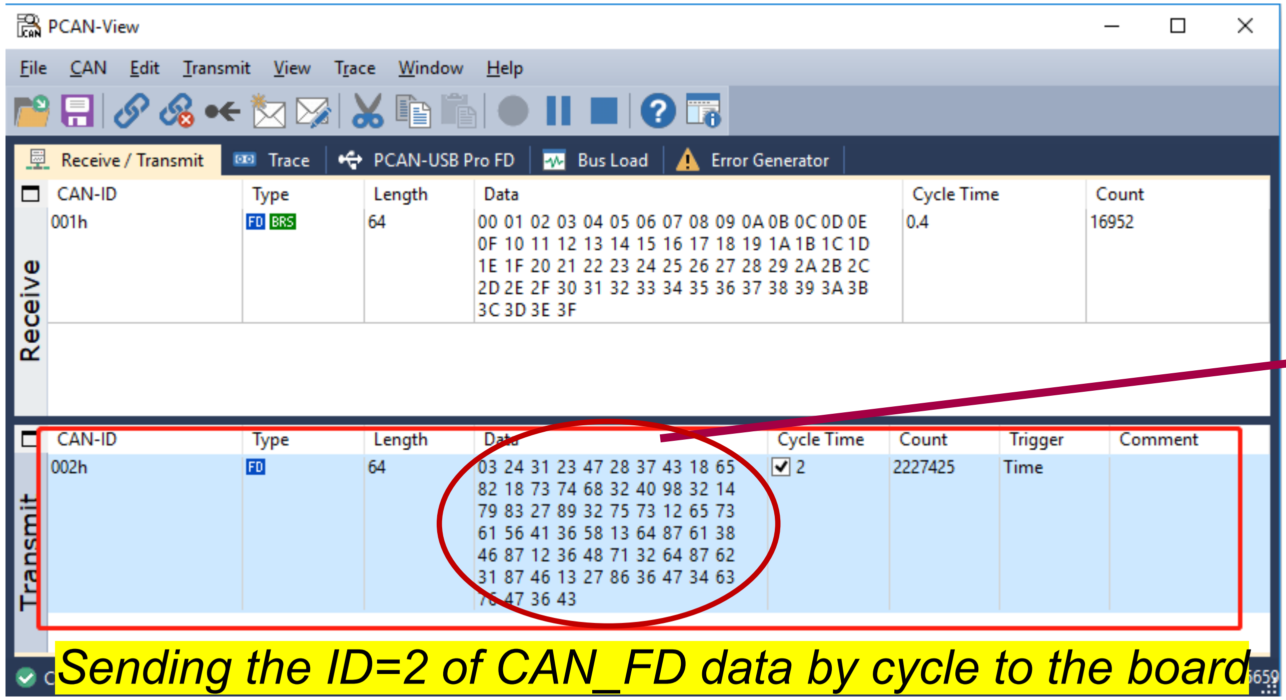
Receiving the data from CANx port of the board

CAN-ID	Type	Length	Data	Cycle Time	Count	Trigger	Comment
002h	FD	64	03 24 31 23 47 28 37 43 18 65 82 18 73 74 68 32 40 98 32 14 79 83 27 89 32 75 73 12 65 73 61 56 41 36 58 13 64 87 61 38 46 87 12 36 48 71 32 64 87 62 31 87 46 13 27 86 36 47 34 63 76 47 36 43	☐ 2	2226138	Time	

Connected to hardware PCAN-USB Pro FD, Channel 1 | Bit rate: 500 kBit/s / 2 MBit/s | Status: OK | Overruns: 0 | QXmtFull: 5659

Hands-on –CAN_FD: Build and Debug

- **TEST: Receiving data via CAN :**
  1) Open the macro of CAN_RECEIVE, commented out CAN_SEND
  2) Connect the receiving port with PEAK(The test tools we use) through the wire.
  3) Debug the application.

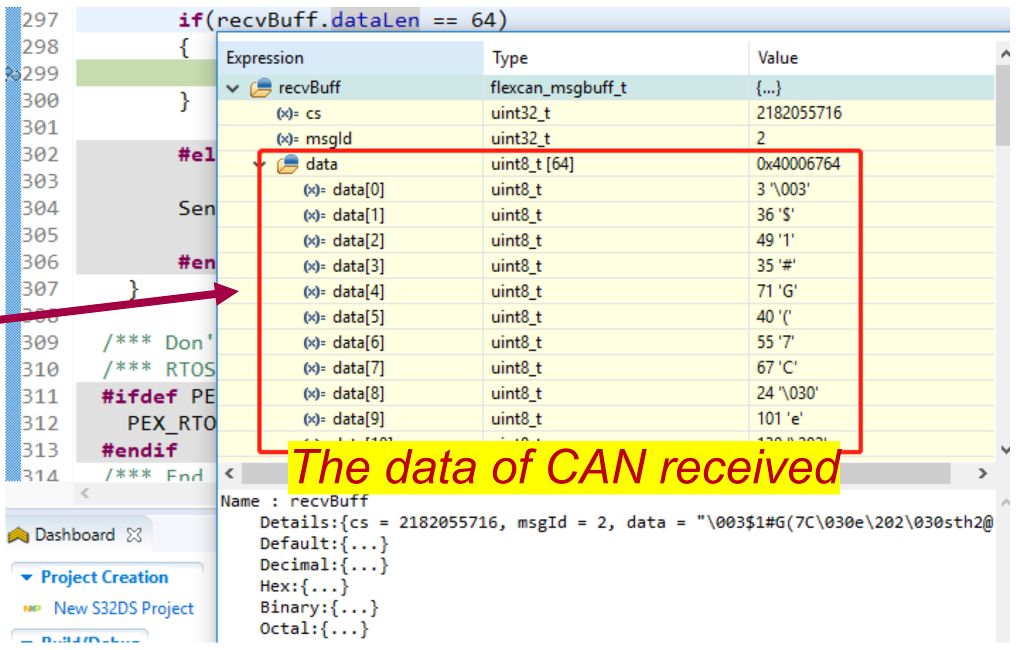


CAN-ID	Type	Length	Data	Cycle Time	Count
001h	FD BRS	64	00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F 30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F	0.4	16952

CAN-ID	Type	Length	Data	Cycle Time	Count	Trigger	Comment
002h	FD	64	03 24 31 23 47 28 37 43 18 65 82 18 73 74 68 32 40 98 32 14 79 83 27 89 32 75 73 12 65 73 61 56 41 36 58 13 64 87 61 38 46 87 12 36 48 71 32 64 87 62 31 87 46 13 27 86 36 47 34 63 76 47 36 43	✓ 2	2227425	Time	

Sending the ID=2 of CAN_FD data by cycle to the board

Debug window



```
        if(recvBuff.dataLen == 64)
        {

        }

        #el

        Sen

        #en
    }

/*** Don'
/*** RTOS
#ifdef PE
  PEX_RTO
#endif
/*** End
```

Expression	Type	Value
recvBuff	flexcan_msgbuff_t	{...}
cs	uint32_t	2182055716
msgId	uint32_t	2
data	uint8_t [64]	0x40006764
data[0]	uint8_t	3 '\003'
data[1]	uint8_t	36 '$'
data[2]	uint8_t	49 '1'
data[3]	uint8_t	35 '#'
data[4]	uint8_t	71 'G'
data[5]	uint8_t	40 '('
data[6]	uint8_t	55 '7'
data[7]	uint8_t	67 'C'
data[8]	uint8_t	24 '\030'
data[9]	uint8_t	101 'e'

```
Name : recvBuff
    Details:{cs = 2182055716, msgId = 2, data = "\003$1#G(7C\030e\202\030sth2@
    Default:{...}
    Decimal:{...}
    Hex:{...}
    Binary:{...}
    Octal:{...}
```

The data of CAN received

03.

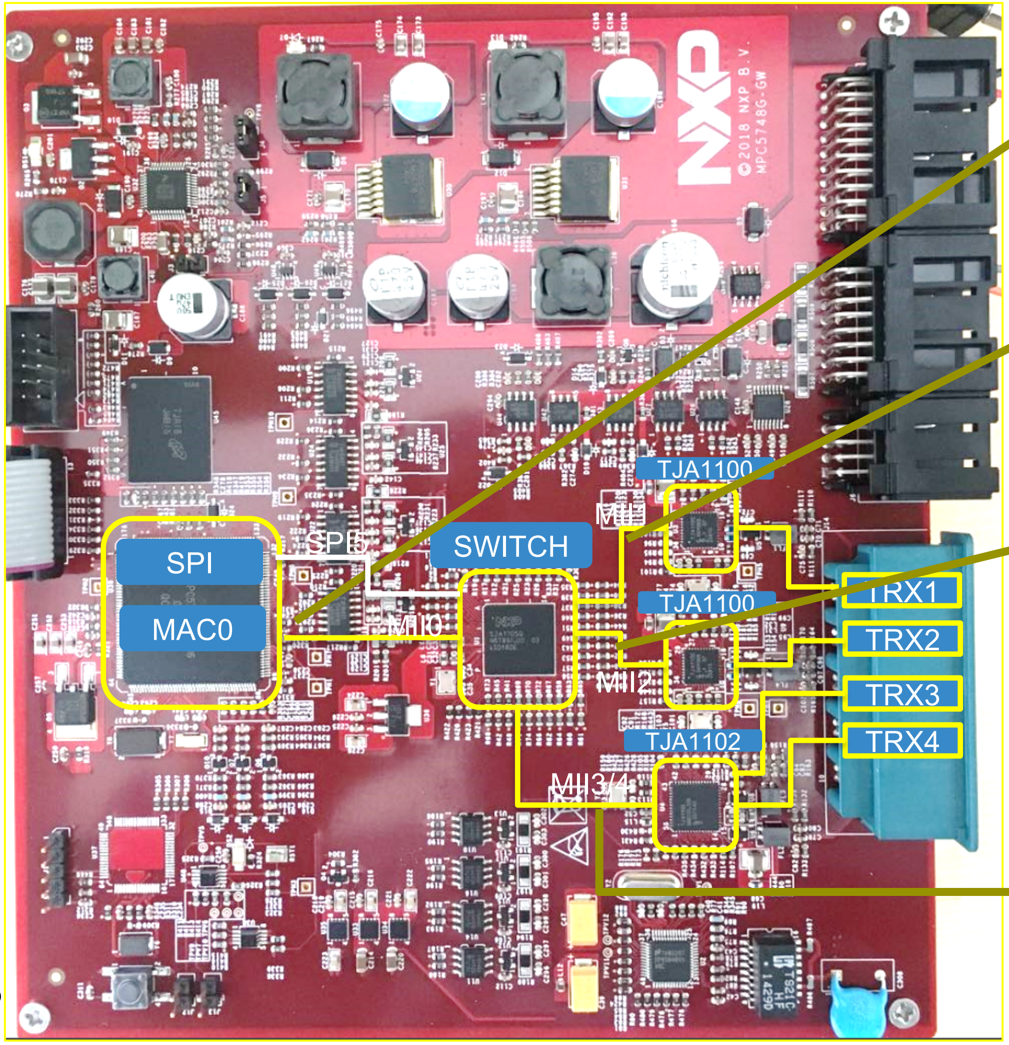
Hands-on – ENET0+SPI

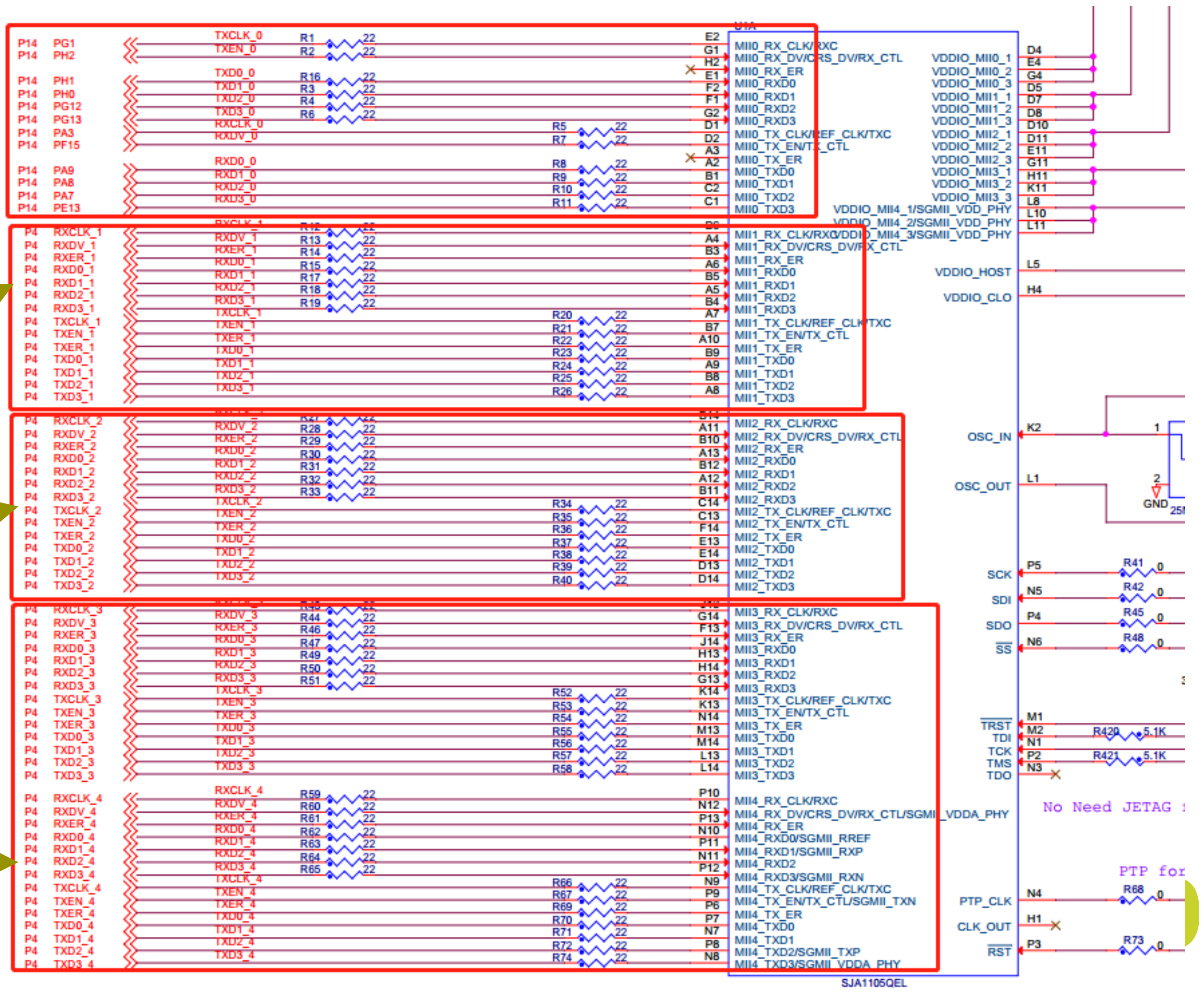
Hands-on – ENET0: Objective

- Features of ENET0 module on MPC5748G
- How to set a pin as output/input with SDK
- How to configure the port of ENET0
- How to configure the SPI module to communicate with Switch(SJA1105)
- How to modify an existing SDK project with S32DS to suit this board
- Use CAN0~CAN7 to send CAN message

Hands-on – ENET0: Resources

- Resources to be used:
  - on-board user ENET ports



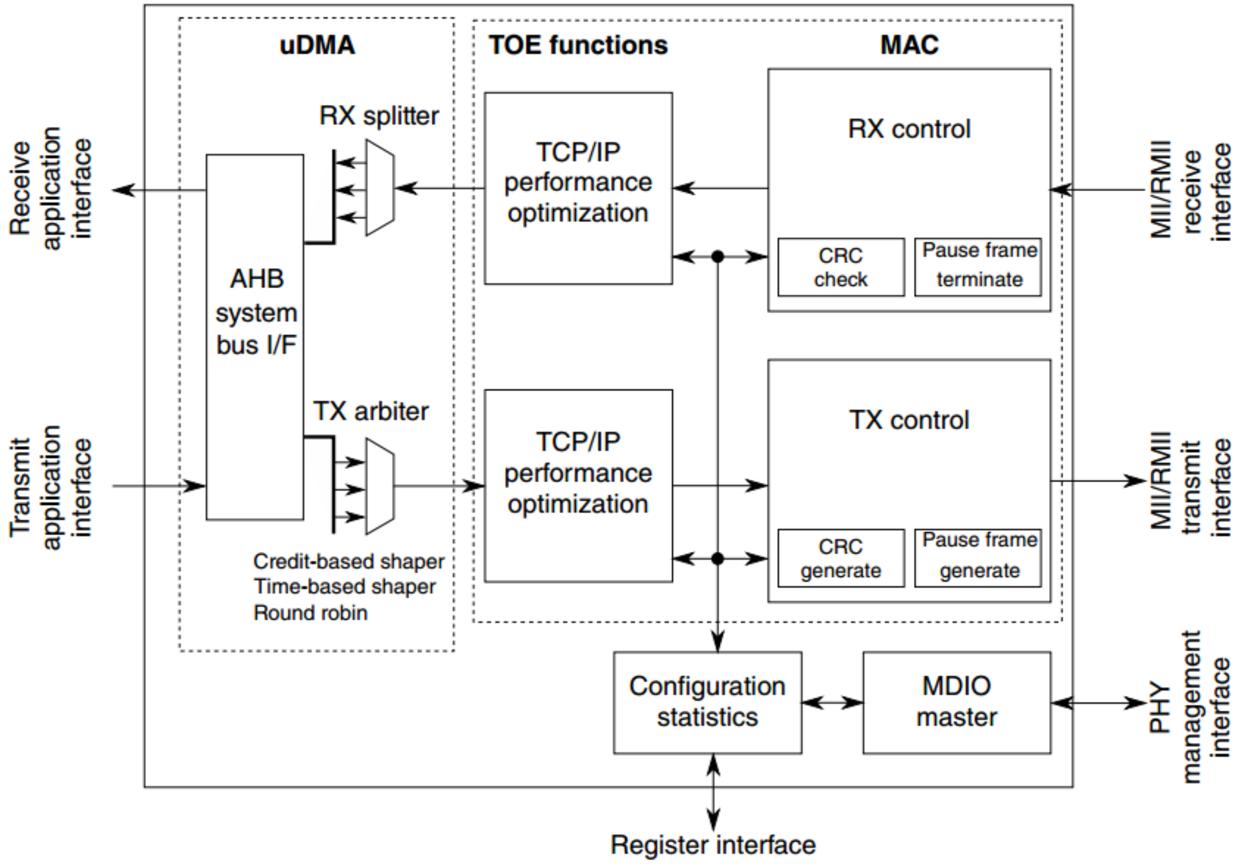


Hands-on – ENET0: ENET-Theory

- The core implements a dual-speed 10/100-Mbit/s Ethernet MAC compliant with the IEEE802.3-2002 standard. The MAC layer provides compatibility with half- or fullduplex 10/100-Mbit/s Ethernet LANs.

- The MAC operation is fully programmable and can be used in Network Interface Card(NIC), bridging, or switching applications. The core implements the remote network monitoring (RMON) counters according to IETF RFC 2819.

- The programmable Ethernet MAC with IEEE 1588 integrates a standard IEEE 802.3 Ethernet MAC with a time-stamping module. The IEEE 1588 standard provides accurate clock synchronization for distributed control nodes for industrial automation applications.

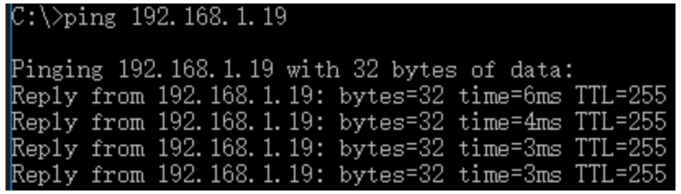


Hands-on – ENET0: SPI-Theory

- Full-duplex, three-wire synchronous transfers
- Master mode
- Slave mode
- Data streaming operation in Slave mode with continuous slave selection
- Buffered transmit operation using the transmit first in first out (TX FIFO) with depth of 4 entries
- Support for 8/16-bit accesses to the PUSH TX FIFO Register Data Field
- Buffered receive operation using the receive FIFO (RX FIFO) with depth of 4 entries
- Asynchronous clocking scheme for Register and Protocol Interfaces
- TX and RX FIFOs can be disabled individually for low-latency updates to SPI queues
- Visibility into TX and RX FIFOs for ease of debugging
- 6 peripheral chip selects (PCSes), expandable to 64 with external demultiplexer
- Deglitching support for up to 32 peripheral chip selects (PCSes) with external demultiplexer

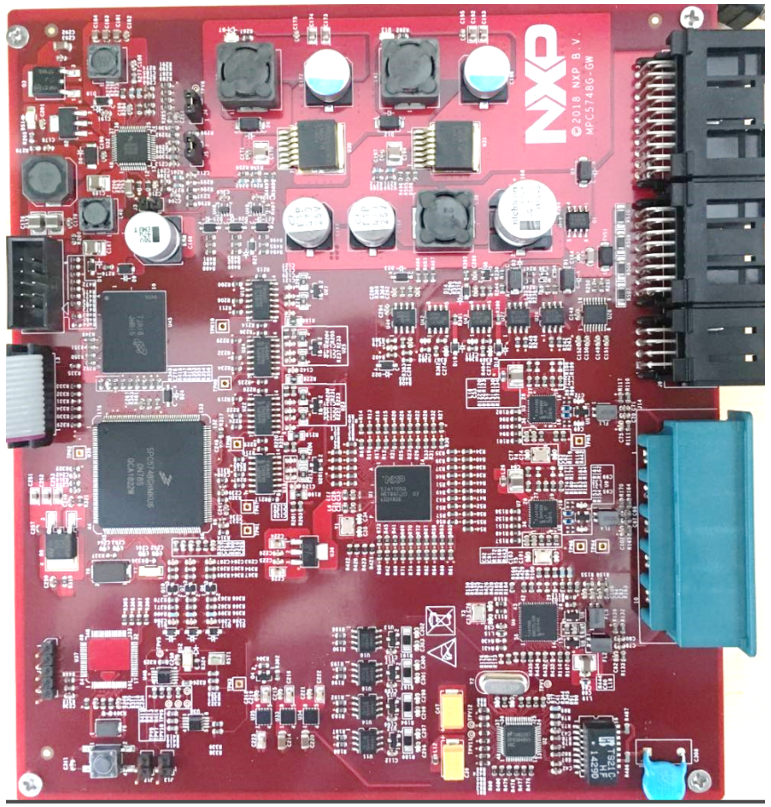
Hands-on – ENET0: Lab Preview

PC terminal



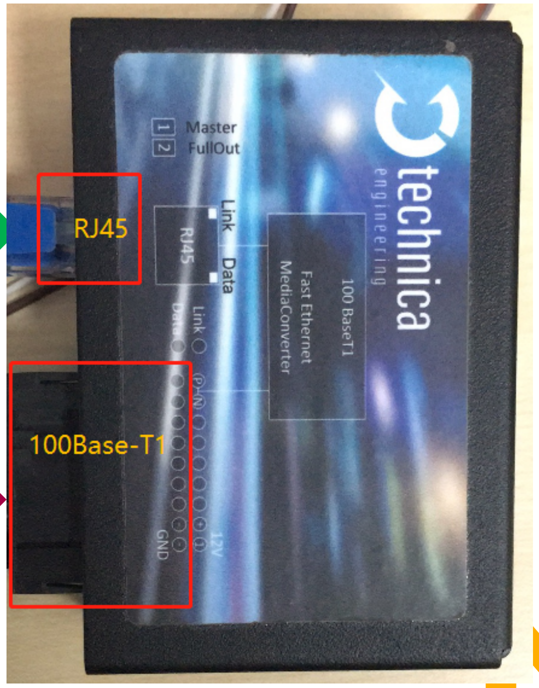
```
C:\>ping 192.168.1.19

Pinging 192.168.1.19 with 32 bytes of data:
Reply from 192.168.1.19: bytes=32 time=6ms TTL=255
Reply from 192.168.1.19: bytes=32 time=4ms TTL=255
Reply from 192.168.1.19: bytes=32 time=3ms TTL=255
Reply from 192.168.1.19: bytes=32 time=3ms TTL=255
```

Gateway Board



ENET

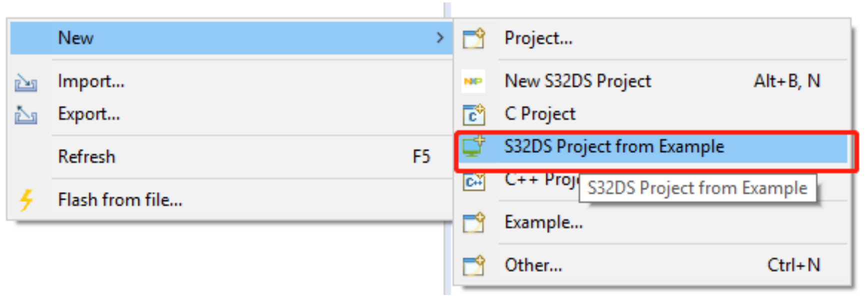
MediaConverter_TJA1100

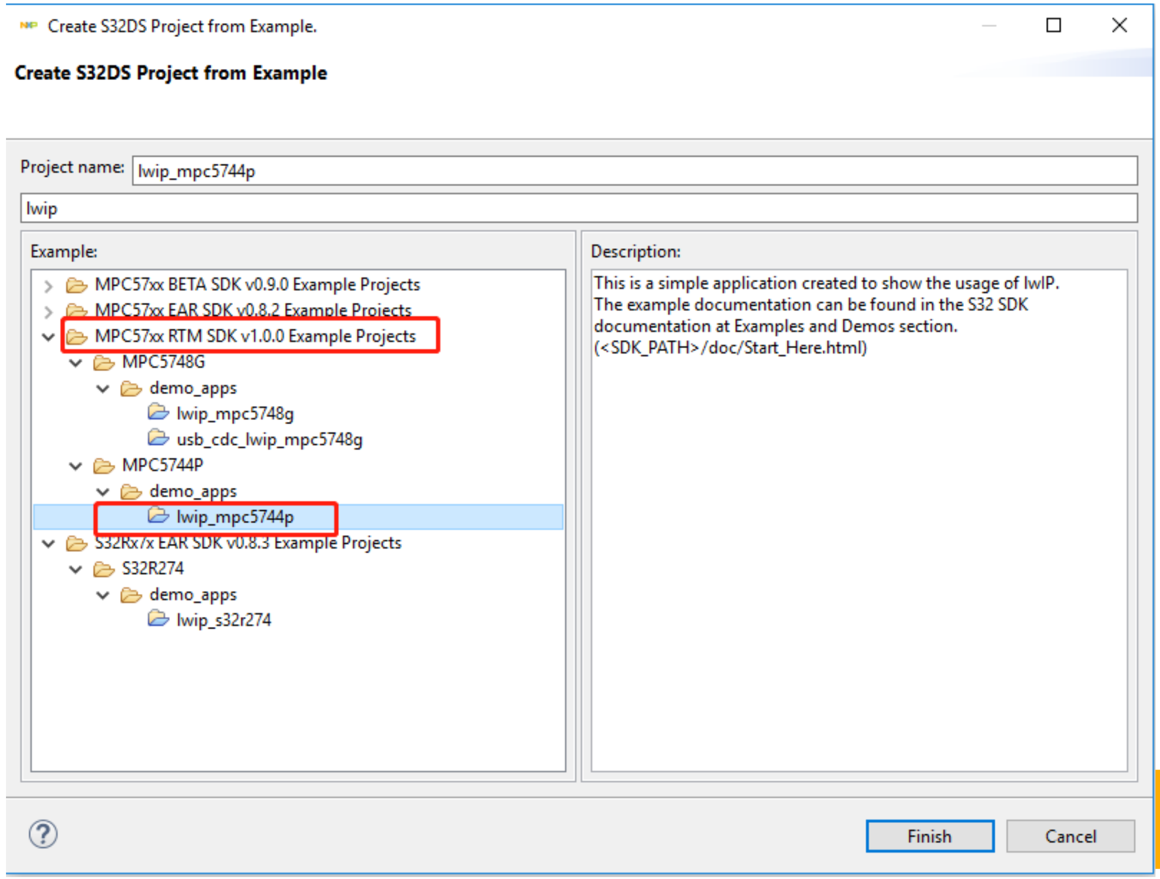


RJ45 <----> 100BASE-T1

Hands-on – ENET0: Import Example Project

- Import ‘enet_ping’ example provided with the SDK:
  - File->New->New S32DS Project from Example
  - Select: **lwip_mpc5748g** from **MPC57xxRTM SDK v1.0.0 Example Projects**




Create S32DS Project from Example.

Create S32DS Project from Example

Project name: lwip_mpc5744p

lwip

Example:

```
> MPC57xx BETA SDK v0.9.0 Example Projects
> MPC57xx EAR SDK v0.8.2 Example Projects
v MPC57xx RTM SDK v1.0.0 Example Projects
  v MPC5748G
    v demo_apps
        lwip_mpc5748g
        usb_cdc_lwip_mpc5748g
  v MPC5744P
    v demo_apps
        lwip_mpc5744p
v S32Rx7x EAR SDK v0.8.3 Example Projects
  v S32R274
    v demo_apps
        lwip_s32r274
```

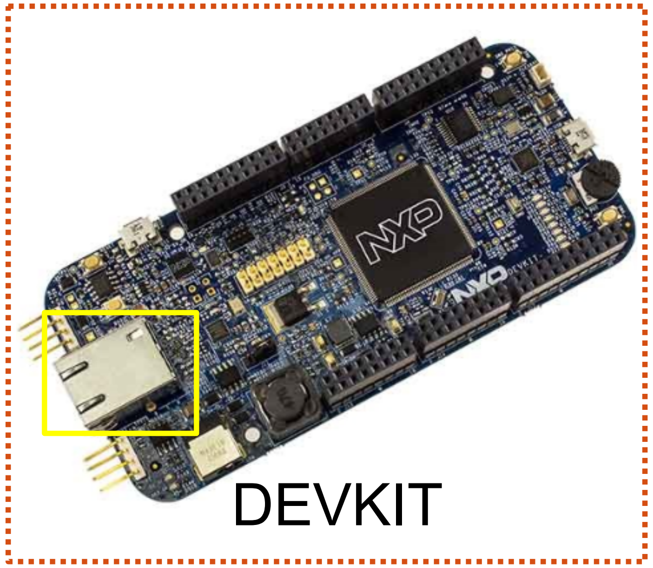
Description:

This is a simple application created to show the usage of lwIP. The example documentation can be found in the S32 SDK documentation at Examples and Demos section. (<SDK_PATH>/doc/Start_Here.html)

Finish Cancel

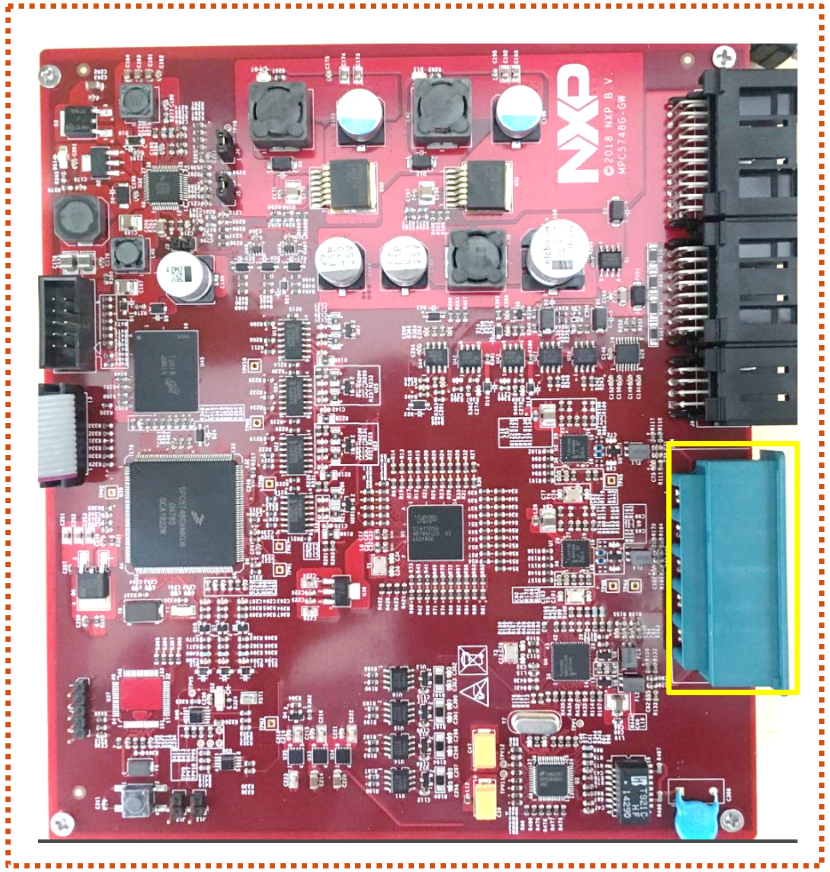
Hands-on – ENET0: Modify

- The example of flexcan_mpc5748g project is suit to DEVKIT
- How to modify?



Modify

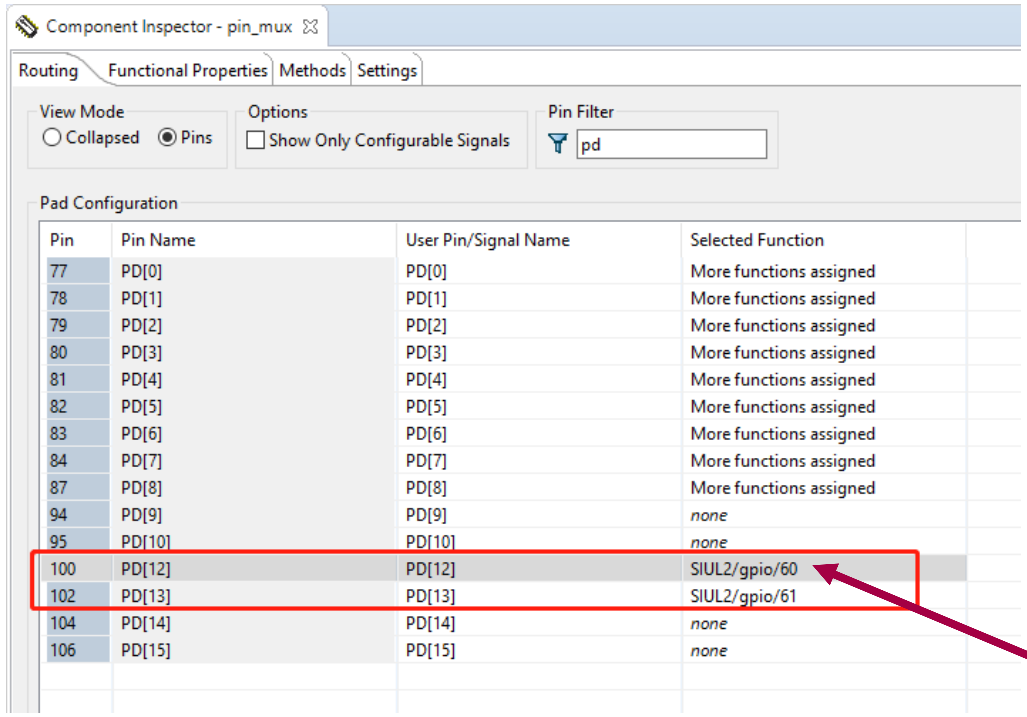
Jobs



1.Peripheral Power Supply
2.SWITCH configuration(Through SPI)
3.ENET configuration
4.Lwip middleware configuration
5.Application code

Hands-on – ENET0: Modify-Peripheral Power Supply

- Enable the peripheral power supply
- Open ‘pin_mux’ component in ‘Component Inspector’ to configure pin routing
- SIUL2 tab -> GPIO 60 （61） > select the pin (one option) + direction output



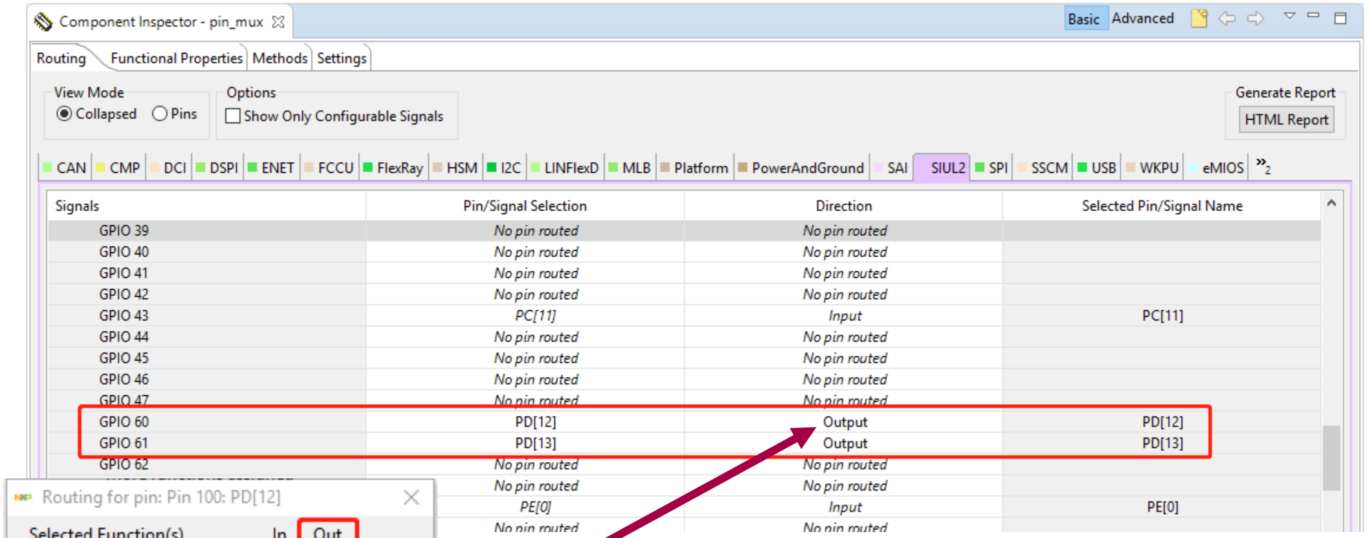
Component Inspector - pin_mux

Routing | Functional Properties | Methods | Settings

View Mode: Collapsed [✓ Pins]
Options: ☐ Show Only Configurable Signals
Pin Filter: pd

Pad Configuration

Pin	Pin Name	User Pin/Signal Name	Selected Function
77	PD[0]	PD[0]	More functions assigned
78	PD[1]	PD[1]	More functions assigned
79	PD[2]	PD[2]	More functions assigned
80	PD[3]	PD[3]	More functions assigned
81	PD[4]	PD[4]	More functions assigned
82	PD[5]	PD[5]	More functions assigned
83	PD[6]	PD[6]	More functions assigned
84	PD[7]	PD[7]	More functions assigned
87	PD[8]	PD[8]	More functions assigned
94	PD[9]	PD[9]	none
95	PD[10]	PD[10]	none
100	PD[12]	PD[12]	SIUL2/gpio/60
102	PD[13]	PD[13]	SIUL2/gpio/61
104	PD[14]	PD[14]	none
106	PD[15]	PD[15]	none



Component Inspector - pin_mux

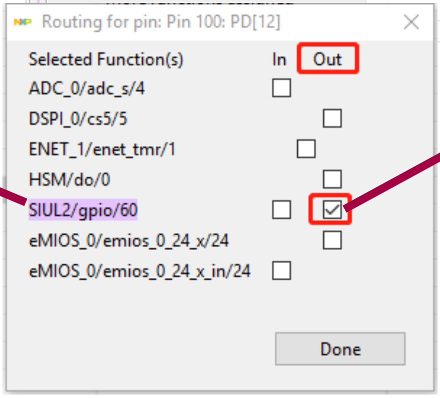
Basic Advanced

Routing | Functional Properties | Methods | Settings

View Mode: [✓ Collapsed] Pins
Options: ☐ Show Only Configurable Signals
Generate Report: HTML Report

CAN | CMP | DCI | DSPI | ENET | FCCU | FlexRay | HSM | I2C | LINFlexD | MLB | Platform | PowerAndGround | SAI | SIUL2 | SPI | SSCM | USB | WKPU | eMIOS

Signals	Pin/Signal Selection	Direction	Selected Pin/Signal Name
GPIO 39	No pin routed	No pin routed	
GPIO 40	No pin routed	No pin routed	
GPIO 41	No pin routed	No pin routed	
GPIO 42	No pin routed	No pin routed	
GPIO 43	PC[11]	Input	PC[11]
GPIO 44	No pin routed	No pin routed	
GPIO 45	No pin routed	No pin routed	
GPIO 46	No pin routed	No pin routed	
GPIO 47	No pin routed	No pin routed	
GPIO 60	PD[12]	Output	PD[12]
GPIO 61	PD[13]	Output	PD[13]
GPIO 62	No pin routed	No pin routed	
	No pin routed	No pin routed	
	PE[0]	Input	PE[0]
	No pin routed	No pin routed	

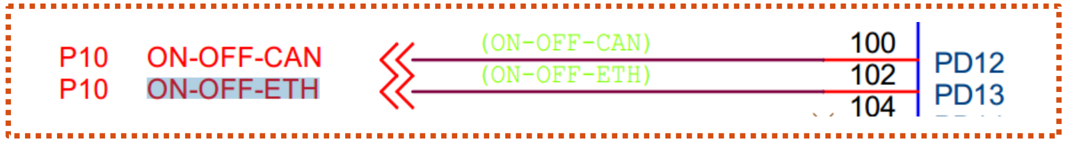


Routing for pin: Pin 100: PD[12]

Selected Function(s)	In	Out
ADC_0/adc_s/4	☐	
DSPI_0/cs5/5		☐
ENET_1/enet_tmr/1	☐	
HSM/do/0		☐
SIUL2/gpio/60	☐	☑
eMIOS_0/emios_0_24_x/24		☐
eMIOS_0/emios_0_24_x_in/24	☐	

Done

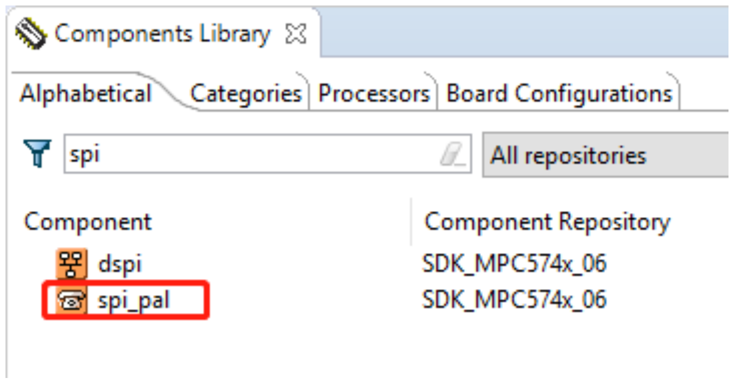
```
void Peripheral_Power_Supply_Init(void)
{
    PINS_DRV_WritePin(PTD, 12, 1);
    PINS_DRV_WritePin(PTD, 13, 1);
}
```



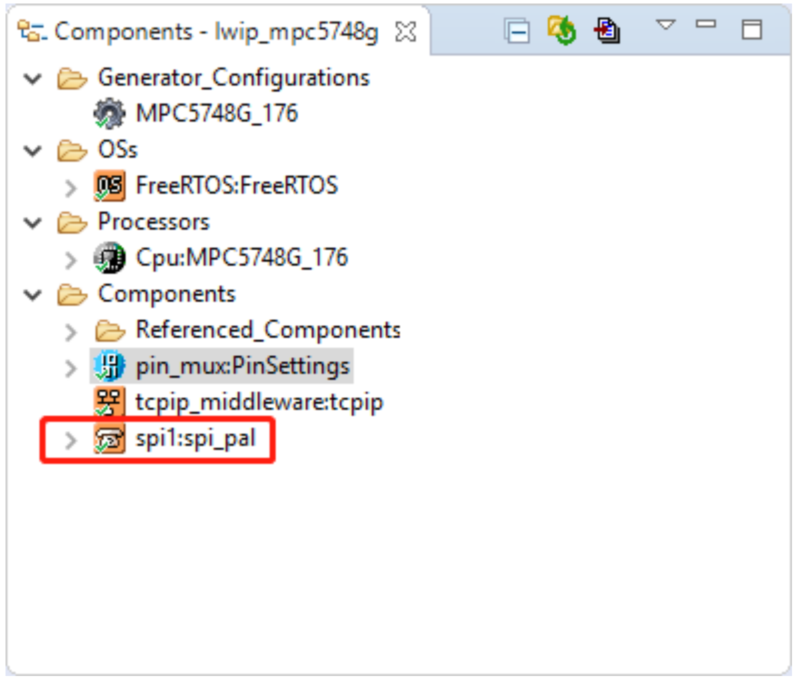
Hands-on – ENET0: SWITCH configuration

(1) Add spi_pal component

- From ‘Components Library’ view, double-click ‘spi_pal’ component to add it the project



Component	Component Repository
dspi	SDK_MPC574x_06
spi_pal	SDK_MPC574x_06

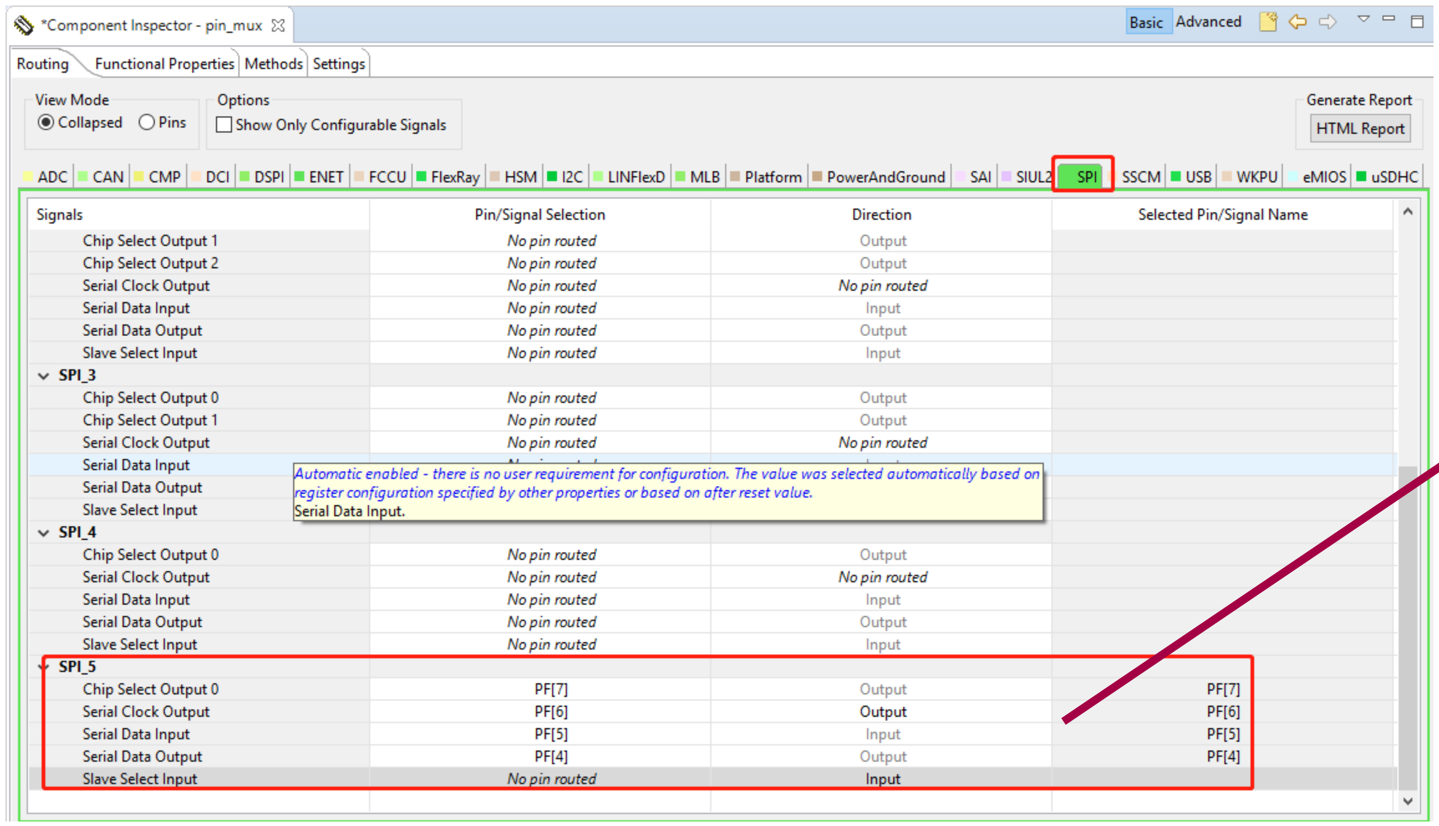


```
Generator_Configurations
    MPC5748G_176
OSs
    FreeRTOS:FreeRTOS
Processors
    Cpu:MPC5748G_176
Components
    Referenced_Components
    pin_mux:PinSettings
    tcpip_middleware:tcpip
    spi1:spi_pal
```

Hands-on – ENET0: SWITCH configuration

(2) The port of SPI configuration

- In ‘pin_mux’ component –> Routing(Collapsed)– select ‘SPI’
  - -Configuration the ports according to the schematic.

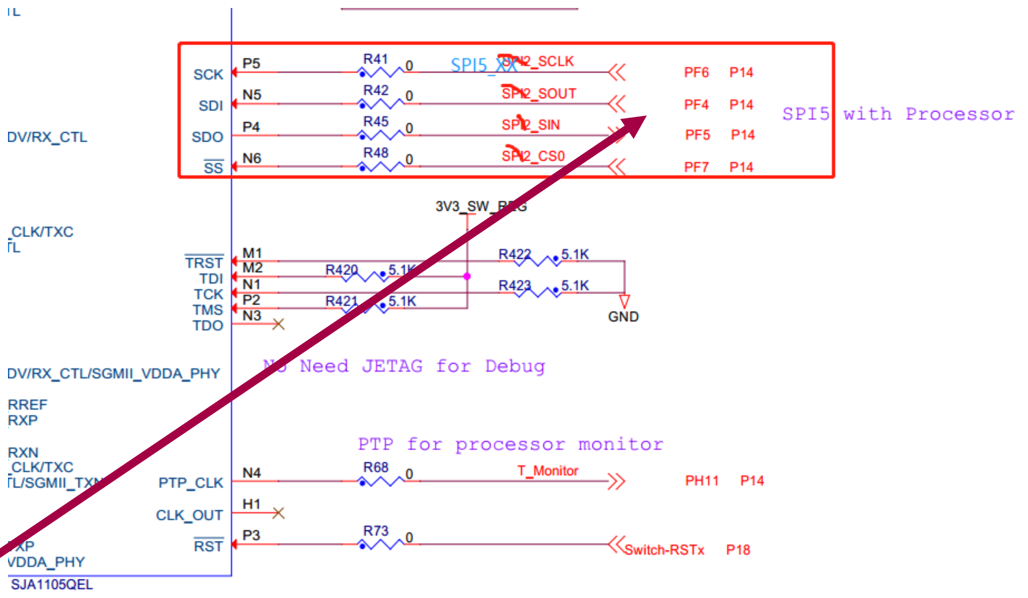


View Mode: [✓ Collapsed] Pins

Options: ☐ Show Only Configurable Signals

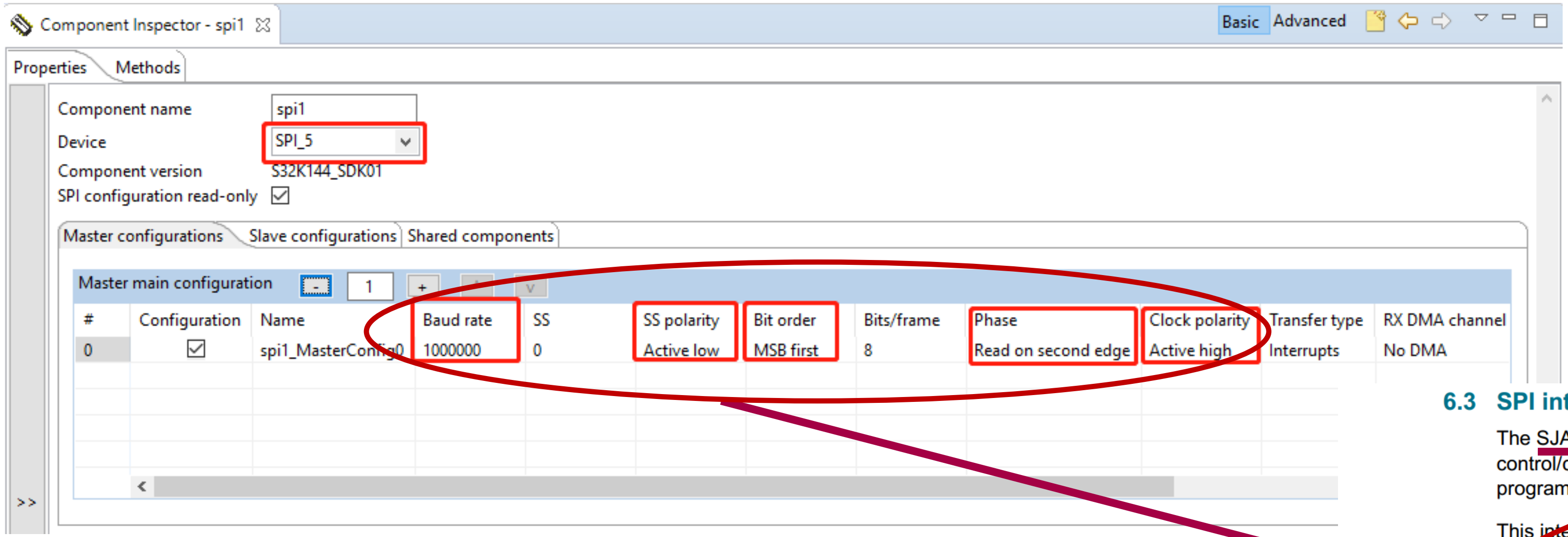
Signals	Pin/Signal Selection	Direction	Selected Pin/Signal Name
Chip Select Output 1	*No pin routed*	Output	
Chip Select Output 2	*No pin routed*	Output	
Serial Clock Output	*No pin routed*	*No pin routed*	
Serial Data Input	*No pin routed*	Input	
Serial Data Output	*No pin routed*	Output	
Slave Select Input	*No pin routed*	Input	
SPI_3			
Chip Select Output 0	*No pin routed*	Output	
Chip Select Output 1	*No pin routed*	Output	
Serial Clock Output	*No pin routed*	*No pin routed*	
Serial Data Input			
Serial Data Output			
Slave Select Input			
SPI_4			
Chip Select Output 0	*No pin routed*	Output	
Serial Clock Output	*No pin routed*	*No pin routed*	
Serial Data Input	*No pin routed*	Input	
Serial Data Output	*No pin routed*	Output	
Slave Select Input	*No pin routed*	Input	
SPI_5			
Chip Select Output 0	PF[7]	Output	PF[7]
Serial Clock Output	PF[6]	Output	PF[6]
Serial Data Input	PF[5]	Input	PF[5]
Serial Data Output	PF[4]	Output	PF[4]
Slave Select Input	*No pin routed*	Input	

Automatic enabled - there is no user requirement for configuration. The value was selected automatically based on register configuration specified by other properties or based on after reset value.
Serial Data Input.



Hands-on – ENET0: SWITCH configuration

(3) SPI component configuration



Component Inspector - spi1

Basic Advanced

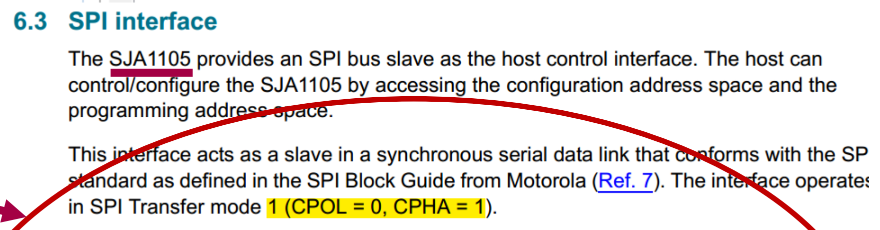
Properties Methods

Component name	spi1
Device	SPI_5
Component version	S32K144_SDK01
SPI configuration read-only	☑

Master configurations | Slave configurations | Shared components

Master main configuration 1

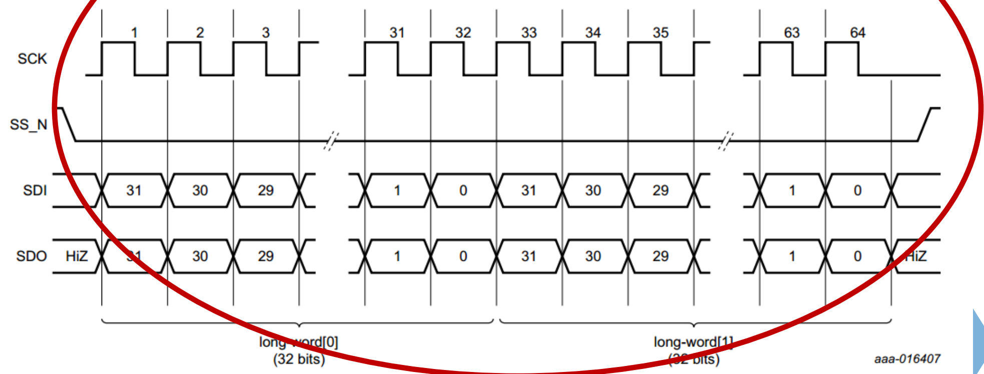
#	Configuration	Name	Baud rate	SS	SS polarity	Bit order	Bits/frame	Phase	Clock polarity	Transfer type	RX DMA channel
0	☑	spi1_MasterConfig0	1000000	0	Active low	MSB first	8	Read on second edge	Active high	Interrupts	No DMA



6.3 SPI interface

The SJA1105 provides an SPI bus slave as the host control interface. The host can control/configure the SJA1105 by accessing the configuration address space and the programming address space.

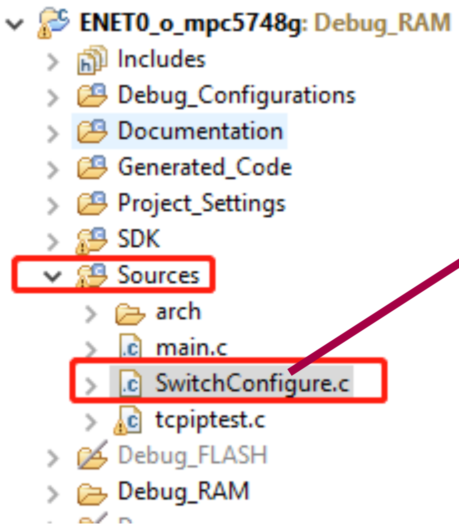
This interface acts as a slave in a synchronous serial data link that conforms with the SP standard as defined in the SPI Block Guide from Motorola (Ref. 7). The interface operates in SPI Transfer mode 1 (CPOL = 0, CPHA = 1).



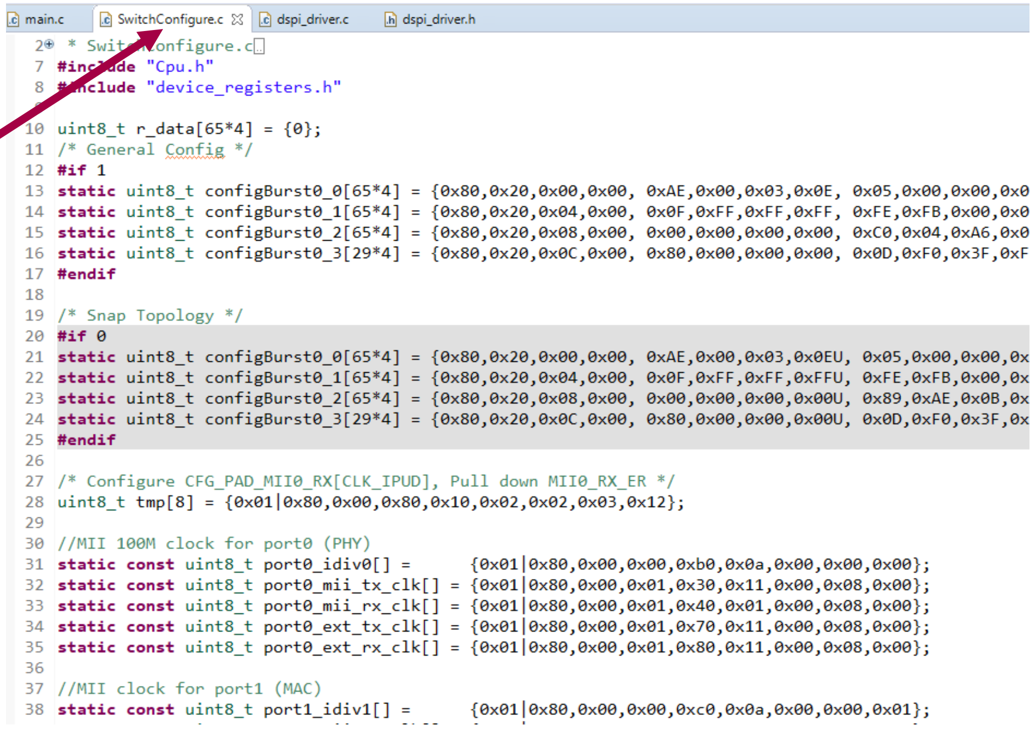
Hands-on – ENET0: SWITCH configuration

(4) Configure SWITCH`s data structure

- We have a tool for configuring SWITCH called *sja1105_tools* (The version of this tool will be updated aperiodically, please contact FAE to get the latest version of the tool).
- You can refer to the sample demo(The project of *ENET0_MPC5748G*) we provided. And copy the *SwitchConfigure.c* file to your project without any modification.



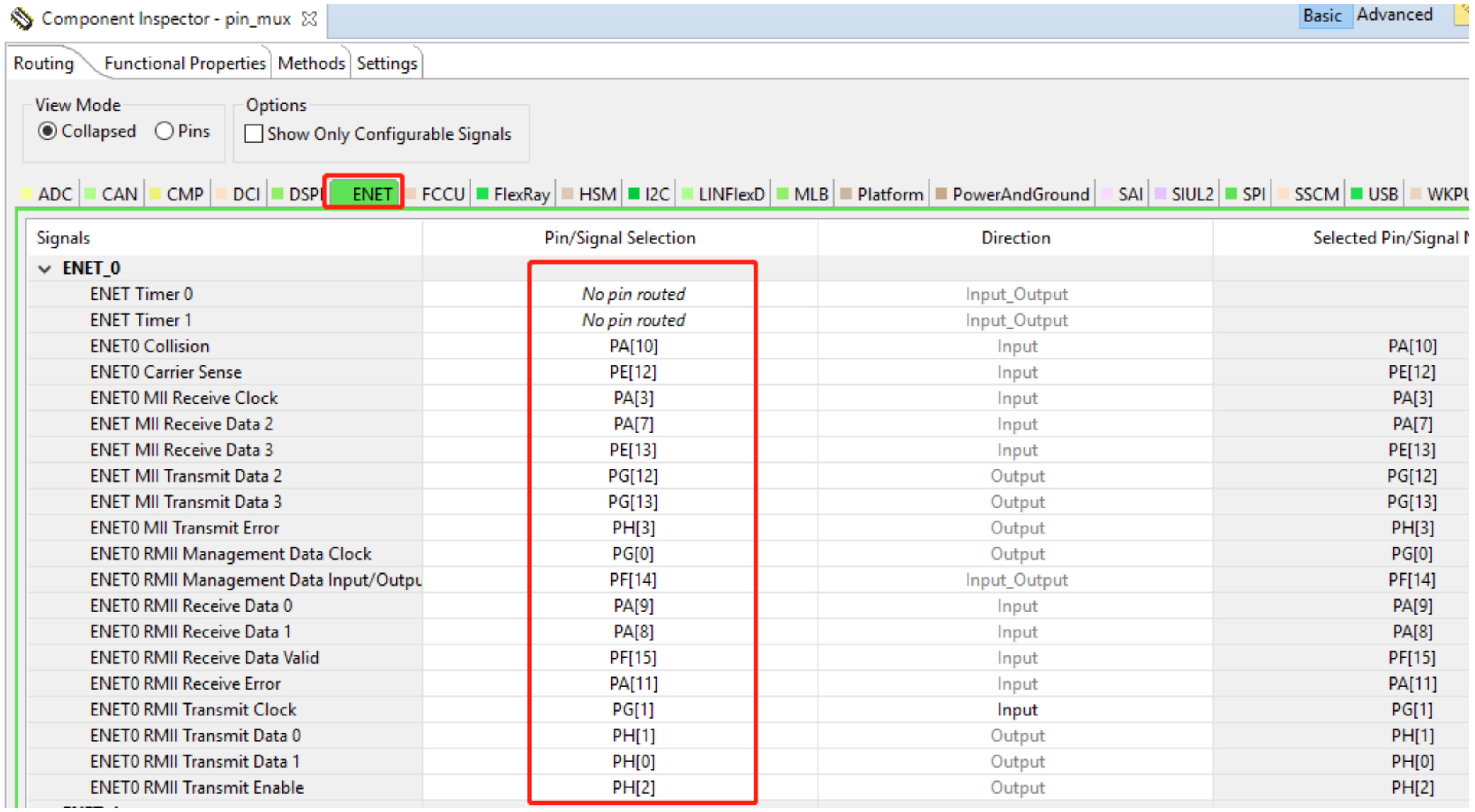
```
ENET0_o_mpc5748g: Debug_RAM
  Includes
  Debug_Configurations
  Documentation
  Generated_Code
  Project_Settings
  SDK
  Sources
    arch
    main.c
    SwitchConfigure.c
    tcpiptest.c
  Debug_FLASH
  Debug_RAM
```



```
 * SwitchConfigure.c
#include "Cpu.h"
#include "device_registers.h"

uint8_t r_data[65*4] = {0};
/* General Config */
#if 1
static uint8_t configBurst0_0[65*4] = {0x80,0x20,0x00,0x00, 0xAE,0x00,0x03,0x0E, 0x05,0x00,0x00,0x0
static uint8_t configBurst0_1[65*4] = {0x80,0x20,0x04,0x00, 0x0F,0xFF,0xFF,0xFF, 0xFE,0xFB,0x00,0x0
static uint8_t configBurst0_2[65*4] = {0x80,0x20,0x08,0x00, 0x00,0x00,0x00,0x00, 0xC0,0x04,0xA6,0x0
static uint8_t configBurst0_3[29*4] = {0x80,0x20,0x0C,0x00, 0x80,0x00,0x00,0x00, 0x0D,0xF0,0x3F,0xF
#endif

/* Snap Topology */
#if 0
static uint8_t configBurst0_0[65*4] = {0x80,0x20,0x00,0x00, 0xAE,0x00,0x03,0x0EU, 0x05,0x00,0x00,0x
static uint8_t configBurst0_1[65*4] = {0x80,0x20,0x04,0x00, 0x0F,0xFF,0xFF,0xFFU, 0xFE,0xFB,0x00,0x
static uint8_t configBurst0_2[65*4] = {0x80,0x20,0x08,0x00, 0x00,0x00,0x00,0x00U, 0x89,0xAE,0x0B,0x
static uint8_t configBurst0_3[29*4] = {0x80,0x20,0x0C,0x00, 0x80,0x00,0x00,0x00U, 0x0D,0xF0,0x3F,0x
#endif

/* Configure CFG_PAD_MII0_RX[CLK_IPUD], Pull down MII0_RX_ER */
uint8_t tmp[8] = {0x01|0x80,0x00,0x80,0x10,0x02,0x02,0x03,0x12};

//MII 100M clock for port0 (PHY)
static const uint8_t port0_idiv0[] =      {0x01|0x80,0x00,0x00,0xb0,0x0a,0x00,0x00,0x00};
static const uint8_t port0_mii_tx_clk[] = {0x01|0x80,0x00,0x01,0x30,0x11,0x00,0x08,0x00};
static const uint8_t port0_mii_rx_clk[] = {0x01|0x80,0x00,0x01,0x40,0x01,0x00,0x08,0x00};
static const uint8_t port0_ext_tx_clk[] = {0x01|0x80,0x00,0x01,0x70,0x11,0x00,0x08,0x00};
static const uint8_t port0_ext_rx_clk[] = {0x01|0x80,0x00,0x01,0x80,0x11,0x00,0x08,0x00};

//MII clock for port1 (MAC)
static const uint8_t port1_idiv1[] =      {0x01|0x80,0x00,0x00,0xc0,0x0a,0x00,0x00,0x01};
```

Hands-on – ENET0: ENET configuration

- In 'pin_mux' component –> Routing(Collapsed)– select 'ENET'
  - -Configuration the ports according to the schematic.

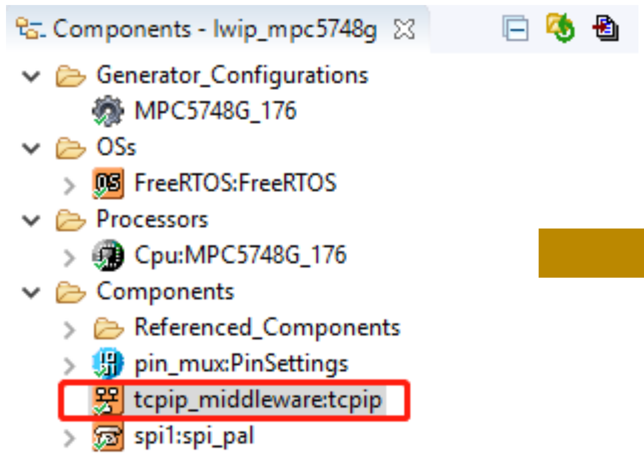


View Mode: (● Collapsed) ○ Pins

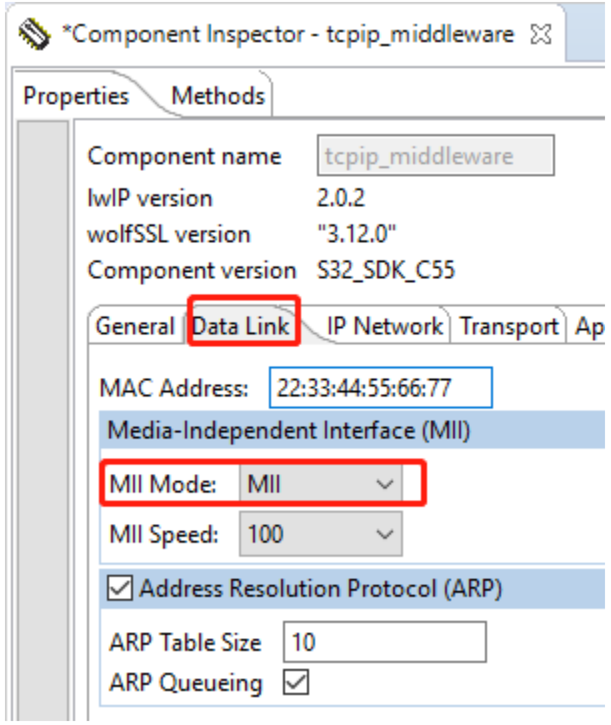
Options: ☐ Show Only Configurable Signals

Signals	Pin/Signal Selection	Direction	Selected Pin/Signal N
ENET_0			
ENET Timer 0	*No pin routed*	Input_Output	
ENET Timer 1	*No pin routed*	Input_Output	
ENET0 Collision	PA[10]	Input	PA[10]
ENET0 Carrier Sense	PE[12]	Input	PE[12]
ENET0 MII Receive Clock	PA[3]	Input	PA[3]
ENET MII Receive Data 2	PA[7]	Input	PA[7]
ENET MII Receive Data 3	PE[13]	Input	PE[13]
ENET MII Transmit Data 2	PG[12]	Output	PG[12]
ENET MII Transmit Data 3	PG[13]	Output	PG[13]
ENET0 MII Transmit Error	PH[3]	Output	PH[3]
ENET0 RMII Management Data Clock	PG[0]	Output	PG[0]
ENET0 RMII Management Data Input/Outpu	PF[14]	Input_Output	PF[14]
ENET0 RMII Receive Data 0	PA[9]	Input	PA[9]
ENET0 RMII Receive Data 1	PA[8]	Input	PA[8]
ENET0 RMII Receive Data Valid	PF[15]	Input	PF[15]
ENET0 RMII Receive Error	PA[11]	Input	PA[11]
ENET0 RMII Transmit Clock	PG[1]	Input	PG[1]
ENET0 RMII Transmit Data 0	PH[1]	Output	PH[1]
ENET0 RMII Transmit Data 1	PH[0]	Output	PH[0]
ENET0 RMII Transmit Enable	PH[2]	Output	PH[2]

Hands-on – ENET0: lwip middleware configuration


Components - lwip_mpc5748g

- Generator_Configurations
  - MPC5748G_176
- OSs
  - FreeRTOS:FreeRTOS
- Processors
  - Cpu:MPC5748G_176
- Components
  - Referenced_Components
  - pin_mux:PinSettings
  - tcpip_middleware:tcpip
  - spi1:spi_pal


*Component Inspector - tcpip_middleware

Properties | Methods

Component name	tcpip_middleware
lwIP version	2.0.2
wolfSSL version	"3.12.0"
Component version	S32_SDK_C55

General | Data Link | IP Network | Transport | Ap

MAC Address: 22:33:44:55:66:77

Media-Independent Interface (MII)

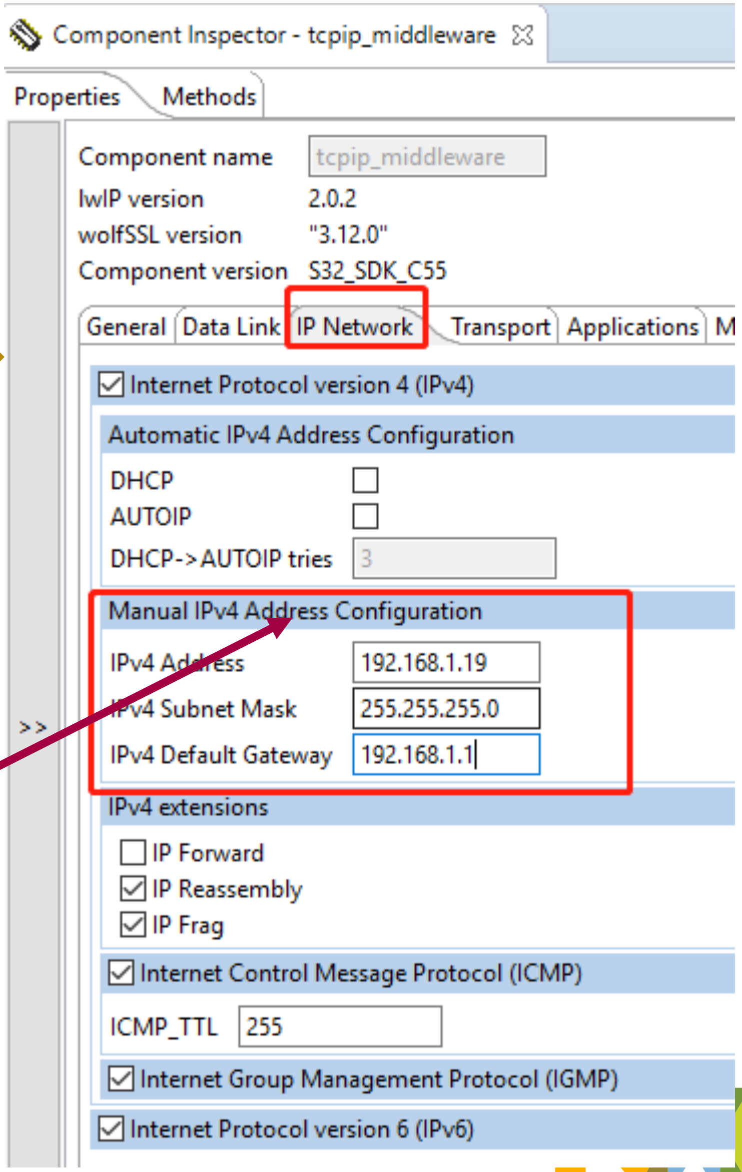
MII Mode: MII

MII Speed: 100

☑ Address Resolution Protocol (ARP)

ARP Table Size: 10

ARP Queueing: ☑


Component Inspector - tcpip_middleware

Properties | Methods

Component name	tcpip_middleware
lwIP version	2.0.2
wolfSSL version	"3.12.0"
Component version	S32_SDK_C55

General | Data Link | IP Network | Transport | Applications | M

☑ Internet Protocol version 4 (IPv4)

Automatic IPv4 Address Configuration

DHCP	☐
AUTOIP	☐
DHCP->AUTOIP tries	3

Manual IPv4 Address Configuration

IPv4 Address	192.168.1.19
IPv4 Subnet Mask	255.255.255.0
IPv4 Default Gateway	192.168.1.1

IPv4 extensions

☐ IP Forward
☑ IP Reassembly
☑ IP Frag

☑ Internet Control Message Protocol (ICMP)

ICMP_TTL: 255

☑ Internet Group Management Protocol (IGMP)

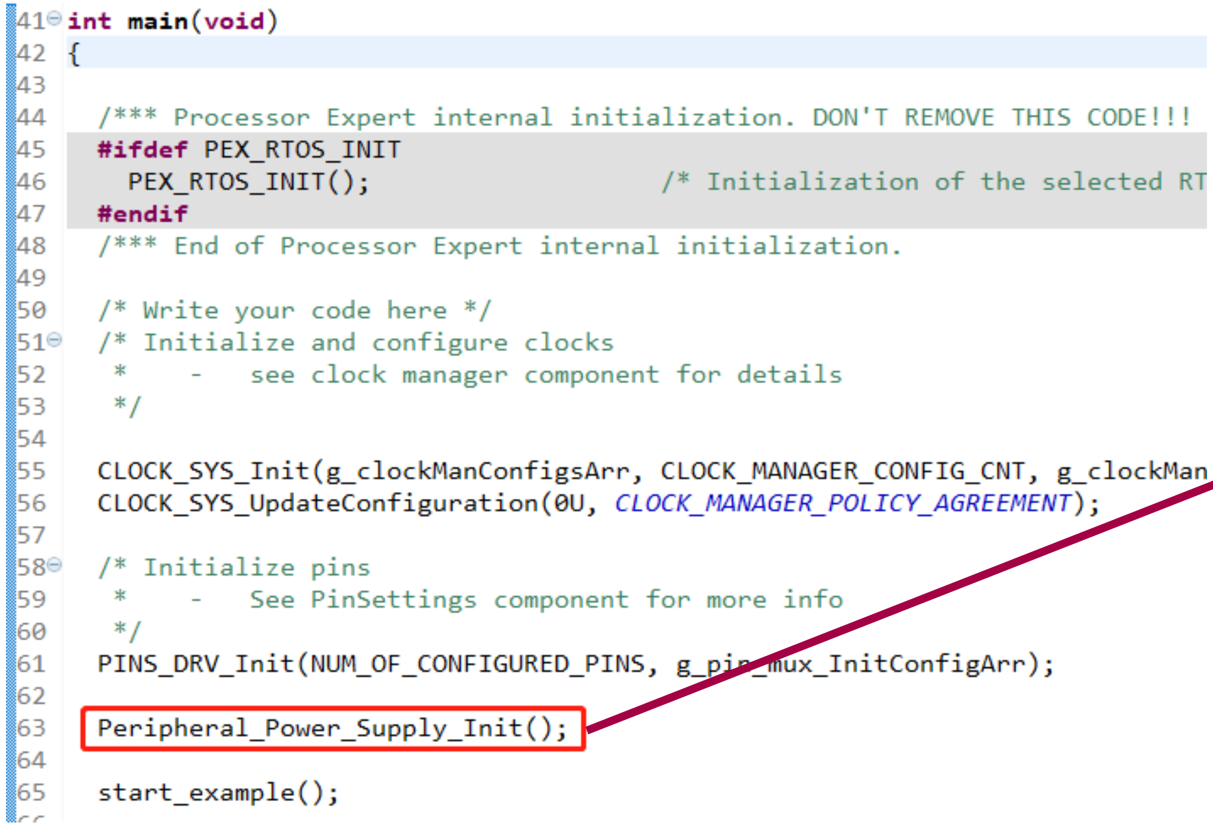
☑ Internet Protocol version 6 (IPv6)

Define it by yourself

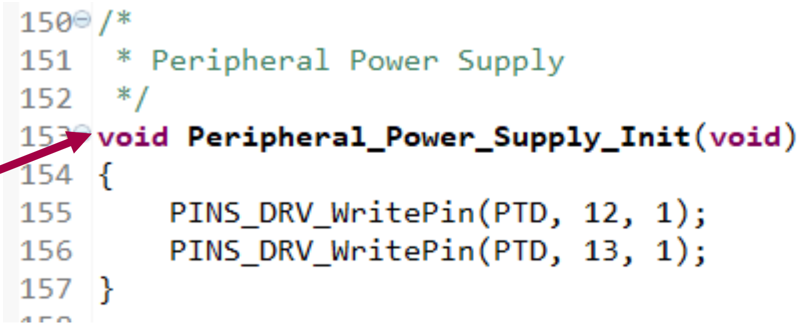
Hands-on – ENET0: Application Code

(1) **Peripheral Power Supply:**

- {Project Name} -> Source -> main.c



```
int main(void)
{

  /*** Processor Expert internal initialization. DON'T REMOVE THIS CODE!!!
  #ifdef PEX_RTOS_INIT
    PEX_RTOS_INIT();                   /* Initialization of the selected RT
  #endif
  /*** End of Processor Expert internal initialization.

  /* Write your code here */
  /* Initialize and configure clocks
   *    -   see clock manager component for details
   */

  CLOCK_SYS_Init(g_clockManConfigsArr, CLOCK_MANAGER_CONFIG_CNT, g_clockMan
  CLOCK_SYS_UpdateConfiguration(0U, CLOCK_MANAGER_POLICY_AGREEMENT);

  /* Initialize pins
   *    -   See PinSettings component for more info
   */
  PINS_DRV_Init(NUM_OF_CONFIGURED_PINS, g_pin_mux_InitConfigArr);

  Peripheral_Power_Supply_Init();

  start_example();
```



```
/*
 * Peripheral Power Supply
 */
void Peripheral_Power_Supply_Init(void)
{
    PINS_DRV_WritePin(PTD, 12, 1);
    PINS_DRV_WritePin(PTD, 13, 1);
}
```

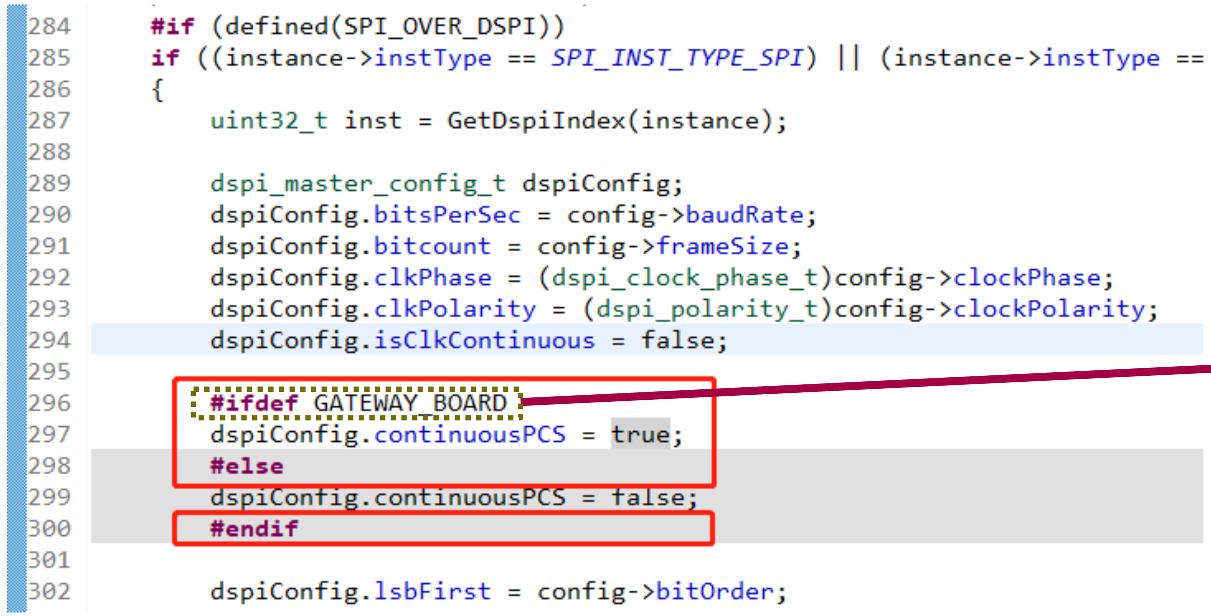
Hands-on – ENET0: Application Code

(2) Modify the configuration of SPI to adaptation SJA1105:

- {Project Name} -> SDK -> platform -> pal -> spi -> src -> spi_pal.c

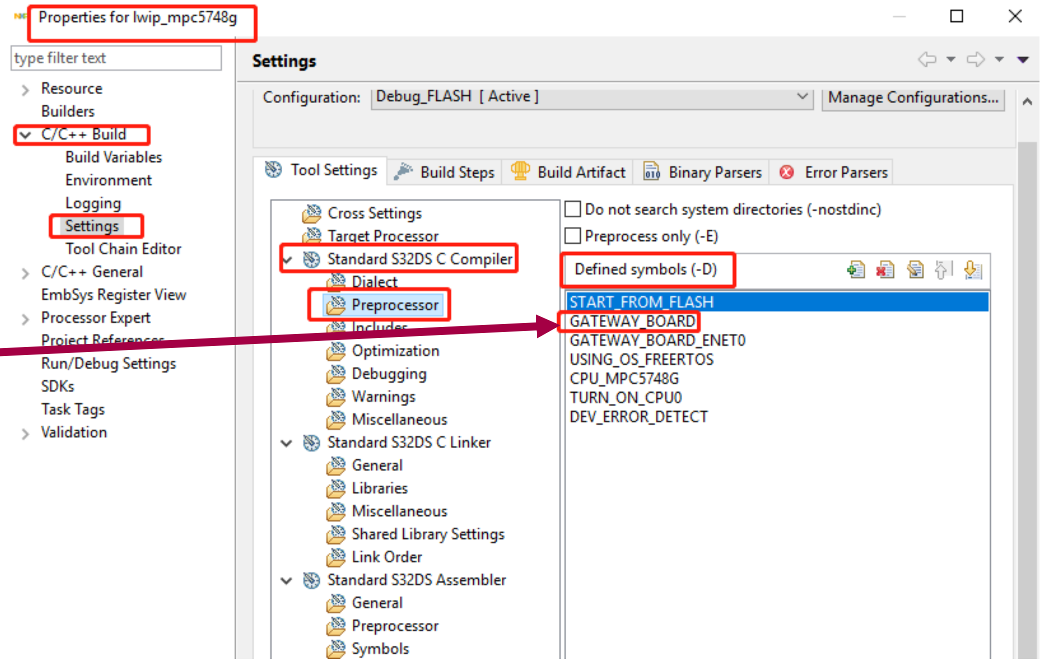
Note:

If you have a patched after update the SDK, you can ignore this modification



```
    #if (defined(SPI_OVER_DSPI))
    if ((instance->instType == SPI_INST_TYPE_SPI) || (instance->instType ==
    {
        uint32_t inst = GetDspiIndex(instance);

        dspi_master_config_t dspiConfig;
        dspiConfig.bitsPerSec = config->baudRate;
        dspiConfig.bitcount = config->frameSize;
        dspiConfig.clkPhase = (dspi_clock_phase_t)config->clockPhase;
        dspiConfig.clkPolarity = (dspi_polarity_t)config->clockPolarity;
        dspiConfig.isClkContinuous = false;

        #ifdef GATEWAY_BOARD
        dspiConfig.continuousPCS = true;
        #else
        dspiConfig.continuousPCS = false;
        #endif

        dspiConfig.lsbFirst = config->bitOrder;
```

Add macro definition



Properties for lwip_mpc5748g

type filter text

- Resource
- Builders
- C/C++ Build
  - Build Variables
  - Environment
  - Logging
  - Settings
  - Tool Chain Editor
- C/C++ General
- EmbSys Register View
- Processor Expert
- Project References
- Run/Debug Settings
- SDKs
- Task Tags
- Validation

Settings

Configuration: Debug_FLASH [Active] Manage Configurations...

Tool Settings | Build Steps | Build Artifact | Binary Parsers | Error Parsers

- Cross Settings
- Target Processor
- Standard S32DS C Compiler
  - Dialect
  - Preprocessor
  - Includes
  - Optimization
  - Debugging
  - Warnings
  - Miscellaneous
- Standard S32DS C Linker
  - General
  - Libraries
  - Miscellaneous
  - Shared Library Settings
  - Link Order
- Standard S32DS Assembler
  - General
  - Preprocessor
  - Symbols

☐ Do not search system directories (-nostdinc)
☐ Preprocess only (-E)

Defined symbols (-D)

```
START_FROM_FLASH
GATEWAY_BOARD
GATEWAY_BOARD_ENET0
USING_OS_FREERTOS
CPU_MPC5748G
TURN_ON_CPU0
DEV_ERROR_DETECT
```

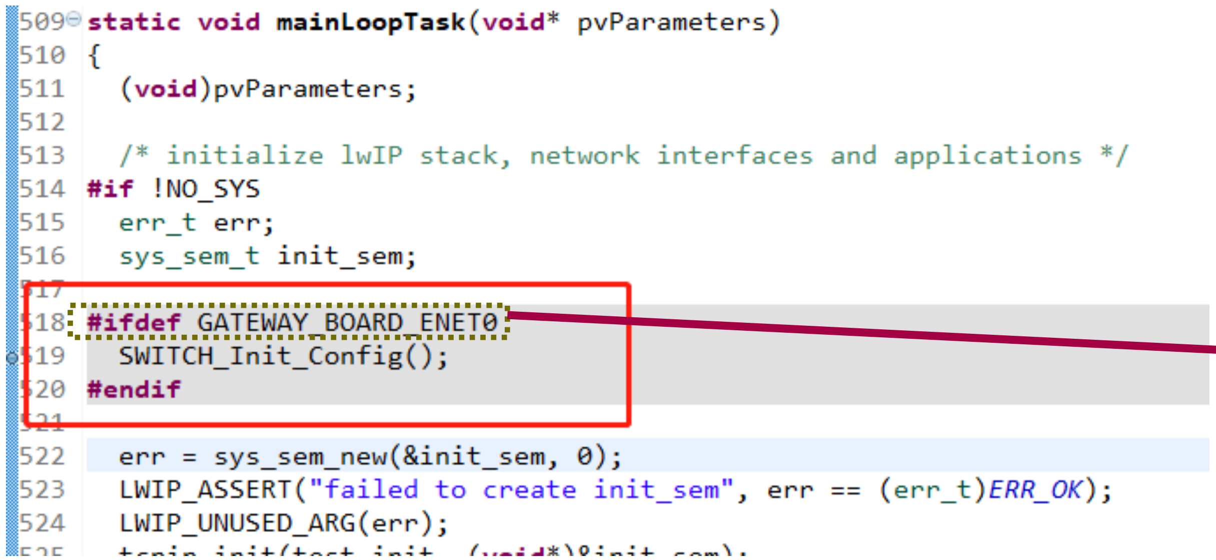
Explanation:

The default configuration of SPI`s CS pin cannot be maintained until the delivery is completed when the send function (*SPI_MasterTransferBlocking()*) is called. In order to match the SWITCH, you need to configure the dspiConfig.continuousPCS = true

Hands-on – ENET0: Application Code

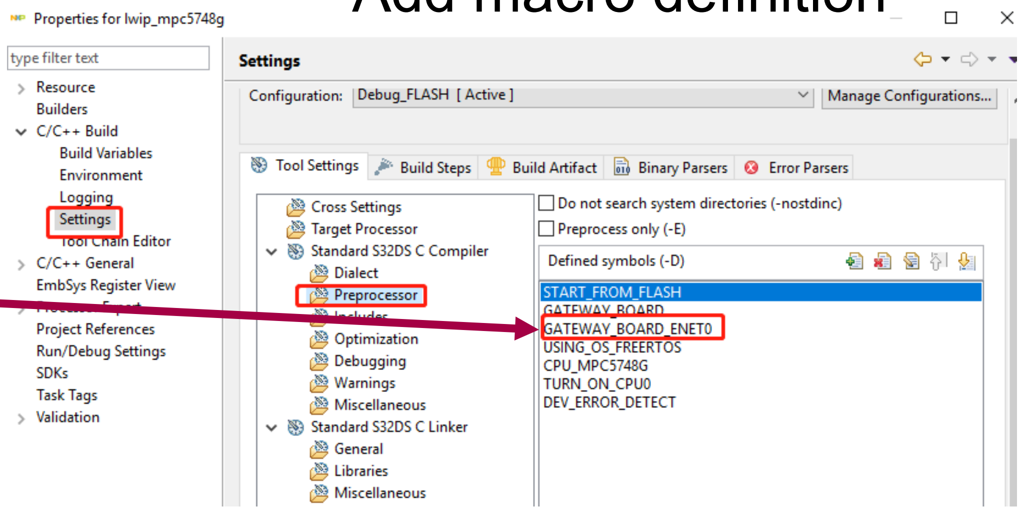
(3) Add the initialization function of the SWITCH:

- {Project Name} -> SDK -> middleware -> tcpip -> tcpip_stack -> demo -> test.c



```
static void mainLoopTask(void* pvParameters)
{
  (void)pvParameters;

  /* initialize lwIP stack, network interfaces and applications */
#if !NO_SYS
  err_t err;
  sys_sem_t init_sem;

#ifdef GATEWAY_BOARD_ENET0
  SWITCH_Init_Config();
#endif

  err = sys_sem_new(&init_sem, 0);
  LWIP_ASSERT("failed to create init_sem", err == (err_t)ERR_OK);
  LWIP_UNUSED_ARG(err);
  tcpip_init(test_init, (void*)&init_sem);
```

Add macro definition



Properties for lwip_mpc5748g

type filter text

- Resource
- Builders
- C/C++ Build
  - Build Variables
  - Environment
  - Logging
  - Settings
  - Tool Chain Editor
- C/C++ General
- EmbSys Register View
- Project References
- Run/Debug Settings
- SDKs
- Task Tags
- Validation

Settings

Configuration: Debug_FLASH [Active] | Manage Configurations...

Tool Settings | Build Steps | Build Artifact | Binary Parsers | Error Parsers

- Cross Settings
- Target Processor
- Standard S32DS C Compiler
  - Dialect
  - Preprocessor
  - Optimization
  - Debugging
  - Warnings
  - Miscellaneous
- Standard S32DS C Linker
  - General
  - Libraries
  - Miscellaneous

☐ Do not search system directories (-nostdinc)

☐ Preprocess only (-E)

Defined symbols (-D)

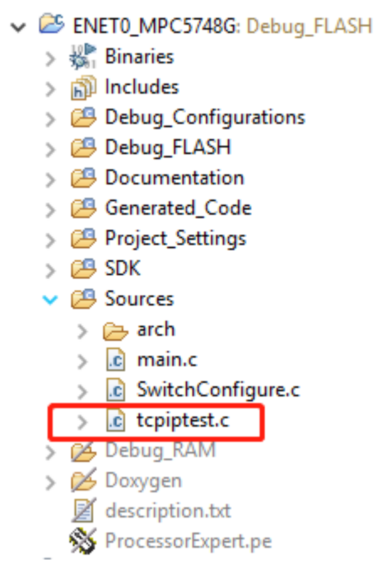
```
START_FROM_FLASH
GATEWAY_BOARD
GATEWAY_BOARD_ENET0
USING_OS_FREERTOS
CPU_MPC5748G
TURN_ON_CPU0
DEV_ERROR_DETECT
```

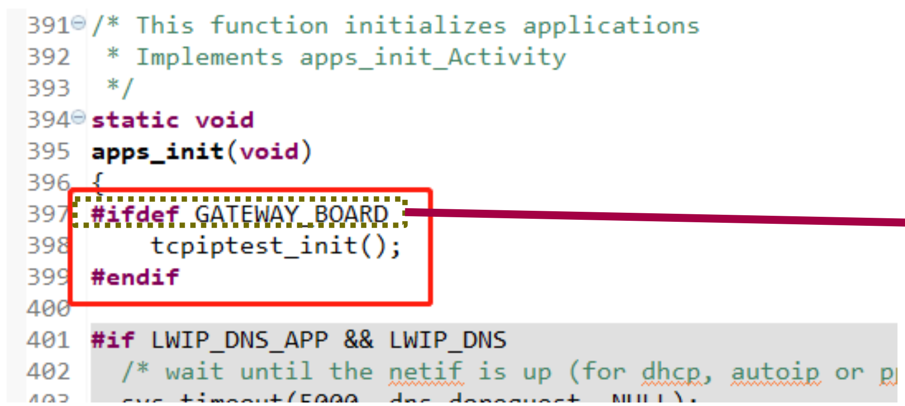
Hands-on – ENET0: Application Code

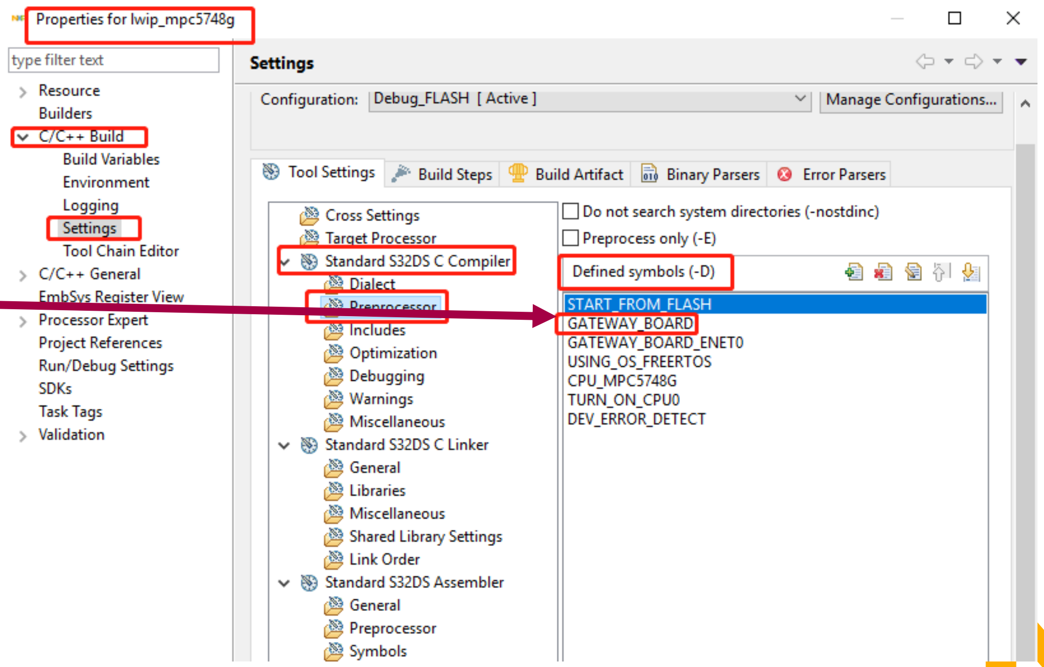
(4) Add client application for lwip_tcp

- You can refer to the sample demo(The project of *ENET0_MPC5748G*) we provided. And copy the *tcpiptest.c* file to your project without any modification.

- Add clicent application call function under the test.c file
  - {Project Name} -> SDK -> middleware -> tcpip -> tcpip_stack -> demo -> test.c


ENET0_MPC5748G: Debug_FLASH
Binaries
Includes
Debug_Configurations
Debug_FLASH
Documentation
Generated_Code
Project_Settings
SDK
Sources
arch
main.c
SwitchConfigure.c
tcpiptest.c
Debug_RAM
Doxygen
description.txt
ProcessorExpert.pe


```
/* This function initializes applications
 * Implements apps_init_Activity
 */
static void
apps_init(void)
{
#ifdef GATEWAY_BOARD
    tcpiptest_init();
#endif

#if LWIP_DNS_APP && LWIP_DNS
  /* wait until the netif is up (for dhcp, autoip or p
```


Properties for lwip_mpc5748g

Settings

Configuration: Debug_FLASH [Active] Manage Configurations...

Resource
Builders
C/C++ Build
Build Variables
Environment
Logging
Settings
Tool Chain Editor
C/C++ General
EmbSys Register View
Processor Expert
Project References
Run/Debug Settings
SDKs
Task Tags
Validation

Tool Settings | Build Steps | Build Artifact | Binary Parsers | Error Parsers

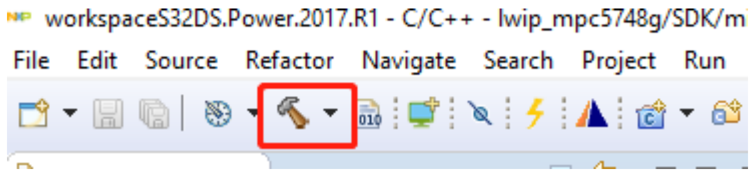
Cross Settings
Target Processor
Standard S32DS C Compiler
Dialect
Preprocessor
Includes
Optimization
Debugging
Warnings
Miscellaneous
Standard S32DS C Linker
General
Libraries
Miscellaneous
Shared Library Settings
Link Order
Standard S32DS Assembler
General
Preprocessor
Symbols

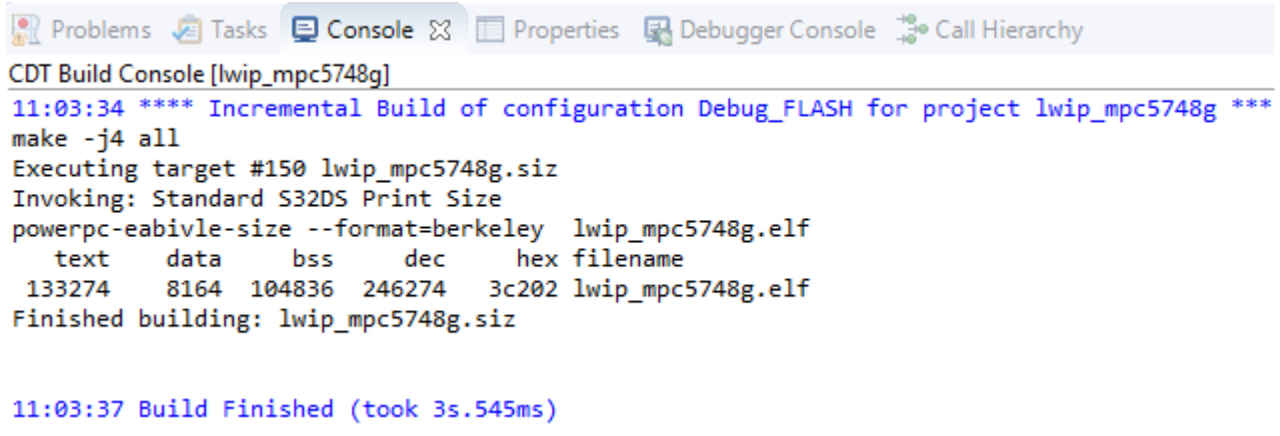
☐ Do not search system directories (-nostdinc)
☐ Preprocess only (-E)

Defined symbols (-D)
START_FROM_FLASH
GATEWAY_BOARD
GATEWAY_BOARD_ENET0
USING_OS_FREERTOS
CPU_MPC5748G
TURN_ON_CPU0
DEV_ERROR_DETECT

Hands-on – ENET0: Build and Debug

- Click the ‘build project’ button – make sure there are no compilation errors

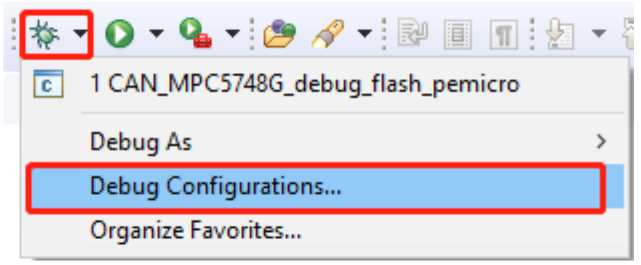

workspaceS32DS.Power.2017.R1 - C/C++ - lwip_mpc5748g/SDK/m
File Edit Source Refactor Navigate Search Project Run

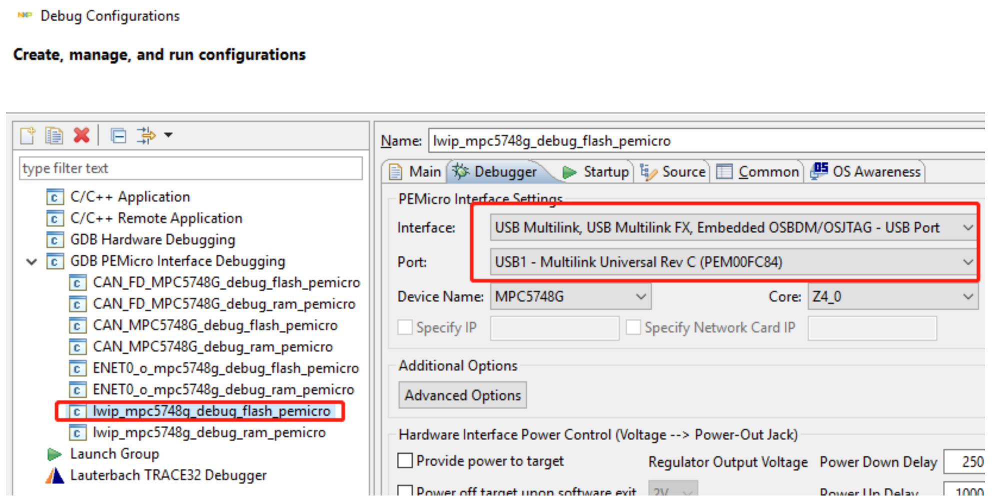

Problems Tasks Console Properties Debugger Console Call Hierarchy

CDT Build Console [lwip_mpc5748g]

```
11:03:34 **** Incremental Build of configuration Debug_FLASH for project lwip_mpc5748g ***
make -j4 all
Executing target #150 lwip_mpc5748g.siz
Invoking: Standard S32DS Print Size
powerpc-eabivle-size --format=berkeley  lwip_mpc5748g.elf
   text    data     bss     dec     hex filename
 133274    8164  104836  246274   3c202 lwip_mpc5748g.elf
Finished building: lwip_mpc5748g.siz


11:03:37 Build Finished (took 3s.545ms)
```

- Select the correct debug configuration and interface to debug the application




Debug Configurations
Create, manage, and run configurations
type filter text
C/C++ Application
C/C++ Remote Application
GDB Hardware Debugging
GDB PEMicro Interface Debugging
CAN_FD_MPC5748G_debug_flash_pemicro
CAN_FD_MPC5748G_debug_ram_pemicro
CAN_MPC5748G_debug_flash_pemicro
CAN_MPC5748G_debug_ram_pemicro
ENET0_o_mpc5748g_debug_flash_pemicro
ENET0_o_mpc5748g_debug_ram_pemicro
lwip_mpc5748g_debug_flash_pemicro
lwip_mpc5748g_debug_ram_pemicro
Launch Group
Lauterbach TRACE32 Debugger
Main Debugger Startup Source Common OS Awareness
PEMicro Interface Settings
Additional Options
Advanced Options
Hardware Interface Power Control (Voltage --> Power-Out Jack)

Name: lwip_mpc5748g_debug_flash_pemicro

Interface: USB Multilink, USB Multilink FX, Embedded OSBDM/OSJTAG - USB Port

Port: USB1 - Multilink Universal Rev C (PEM00FC84)

Device Name: MPC5748G Core: Z4_0

☐ Specify IP ☐ Specify Network Card IP

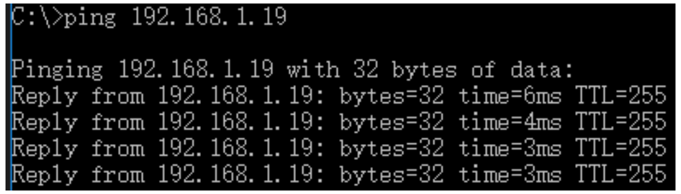
☐ Provide power to target Regulator Output Voltage Power Down Delay 250

☐ Power off target upon software exit 2V Power Up Delay 1000

Hands-on – ENET0: Build and Debug

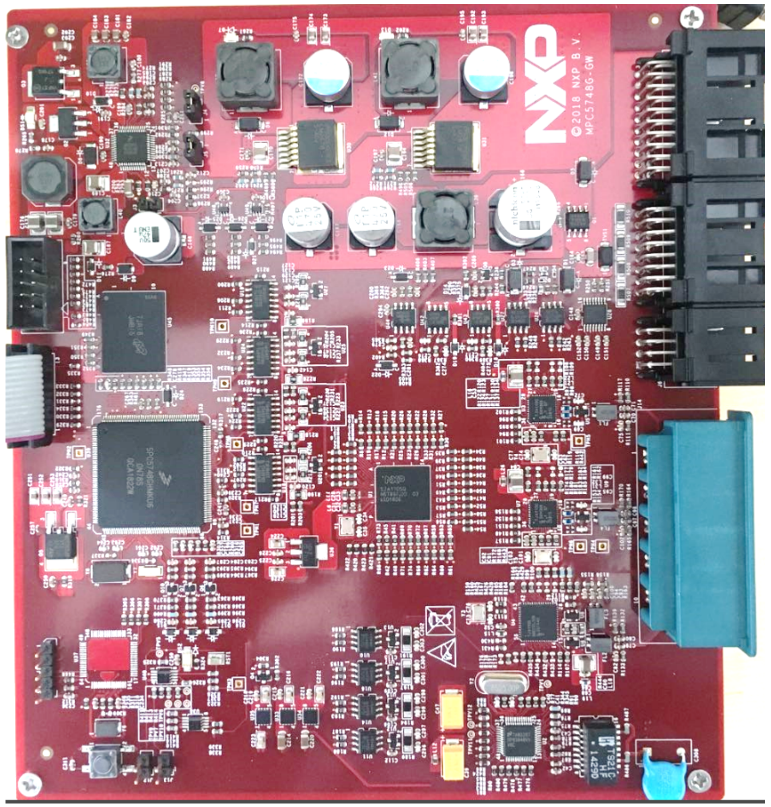
TEST: Ping the board from PC:

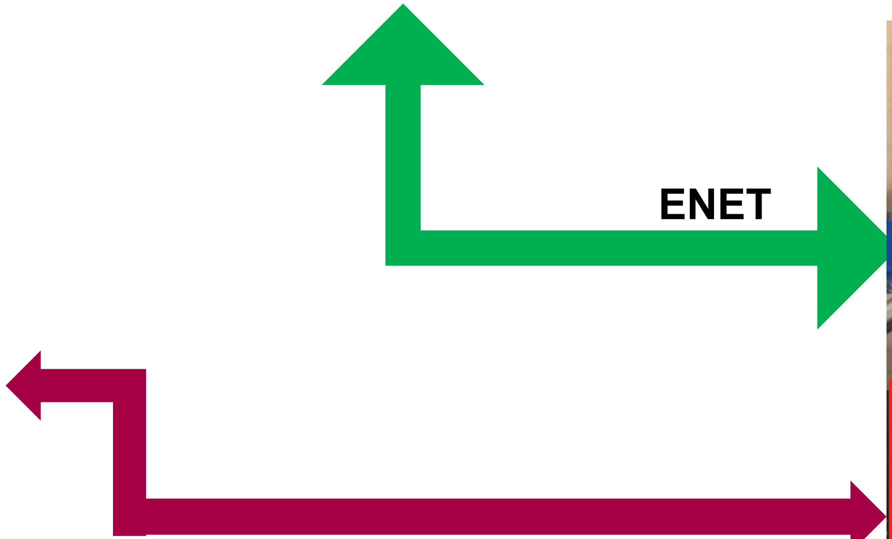
PC terminal



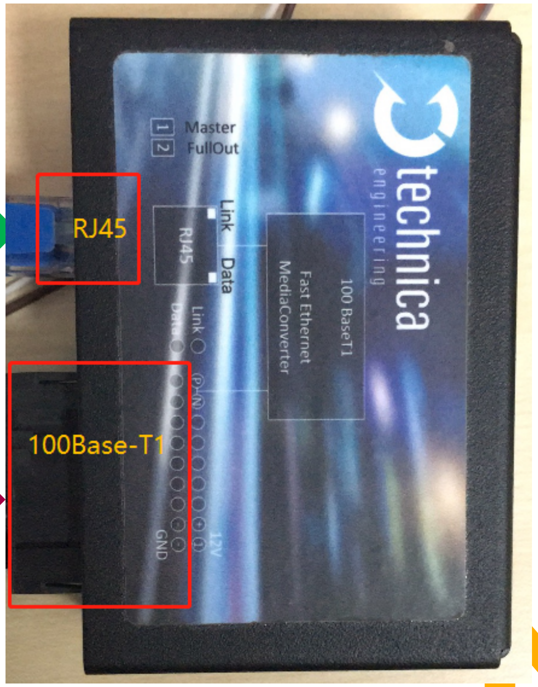
```
C:\>ping 192.168.1.19

Pinging 192.168.1.19 with 32 bytes of data:
Reply from 192.168.1.19: bytes=32 time=6ms TTL=255
Reply from 192.168.1.19: bytes=32 time=4ms TTL=255
Reply from 192.168.1.19: bytes=32 time=3ms TTL=255
Reply from 192.168.1.19: bytes=32 time=3ms TTL=255
```

Gateway Board





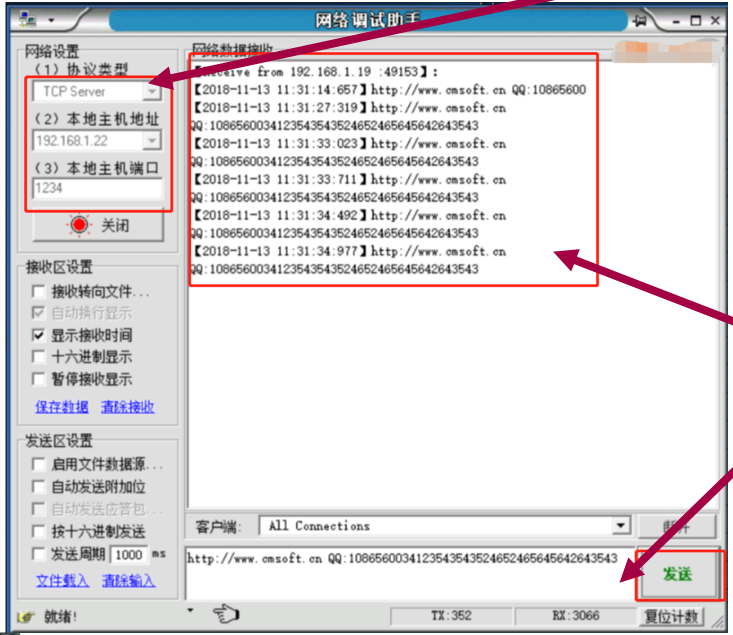
MediaConverter_TJA1100



RJ45 <----> 100BASE-T1

Hands-on – ENET0: Build and Debug

TEST: LWIP_tcp Client:



网络调试助手

网络设置
(1) 协议类型
TCP Server
(2) 本地主机地址
192.168.1.22
(3) 本地主机端口
1234
关闭

接收区设置
- [ ] 接收转向文件...
- [x] 自动换行显示
- [x] 显示接收时间
- [ ] 十六进制显示
- [ ] 暂停接收显示

保存数据 清除接收

发送区设置
- [ ] 启用文件数据源...
- [ ] 自动发送附加位
- [ ] 自动发送应答包...
- [ ] 按十六进制发送
- [ ] 发送周期 1000 ms

文件载入 清除输入

网络数据接收

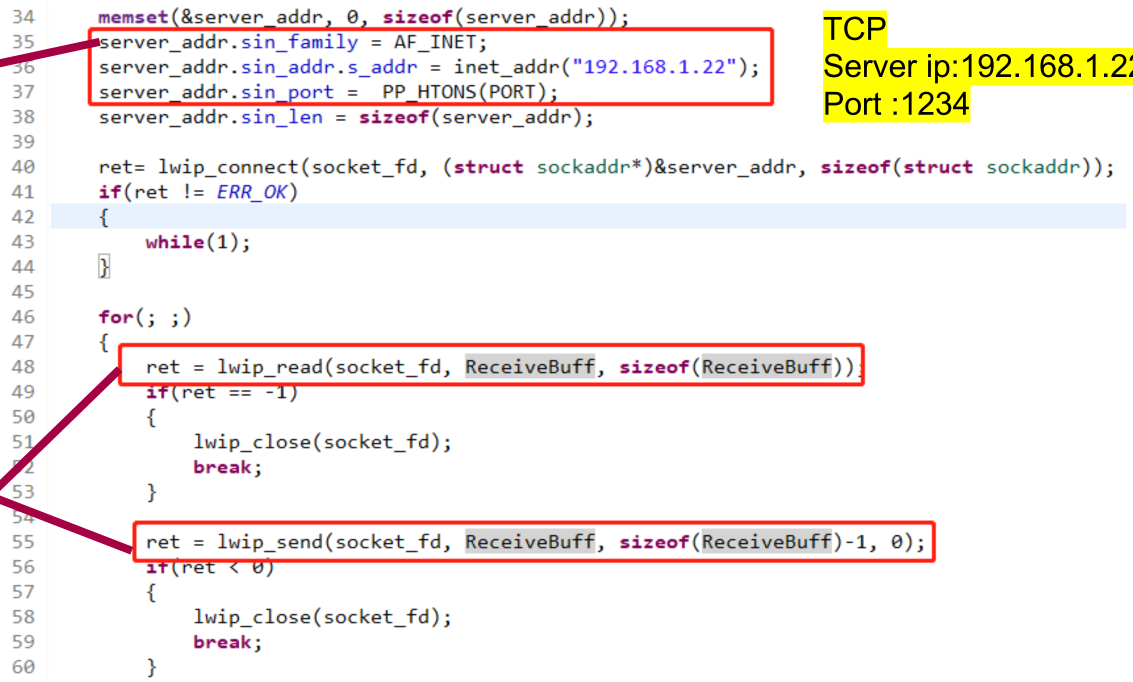
```
【Receive from 192.168.1.19 :49153】:
【2018-11-13 11:31:14:657】http://www.cmsoft.cn QQ:10865600
【2018-11-13 11:31:27:319】http://www.cmsoft.cn
QQ:1086560034123543543524652465645642643543
【2018-11-13 11:31:33:023】http://www.cmsoft.cn
QQ:1086560034123543543524652465645642643543
【2018-11-13 11:31:33:711】http://www.cmsoft.cn
QQ:1086560034123543543524652465645642643543
【2018-11-13 11:31:34:492】http://www.cmsoft.cn
QQ:1086560034123543543524652465645642643543
【2018-11-13 11:31:34:977】http://www.cmsoft.cn
QQ:1086560034123543543524652465645642643543
```

客户端: All Connections 断开

```
http://www.cmsoft.cn QQ:1086560034123543543524652465645642643543
```

发送

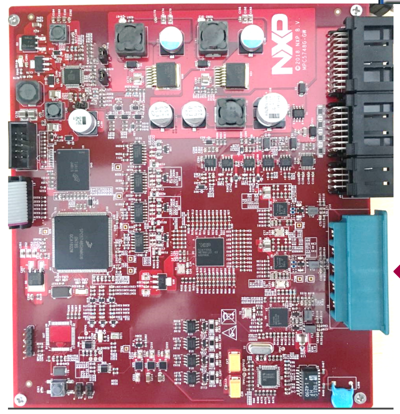
就绪! TX:352 RX:3066 复位计数



```c
memset(&server_addr, 0, sizeof(server_addr));
server_addr.sin_family = AF_INET;
server_addr.sin_addr.s_addr = inet_addr("192.168.1.22");
server_addr.sin_port =  PP_HTONS(PORT);
server_addr.sin_len = sizeof(server_addr);

ret= lwip_connect(socket_fd, (struct sockaddr*)&server_addr, sizeof(struct sockaddr));
if(ret != ERR_OK)
{
    while(1);
}

for(; ;)
{
    ret = lwip_read(socket_fd, ReceiveBuff, sizeof(ReceiveBuff));
    if(ret == -1)
    {
        lwip_close(socket_fd);
        break;
    }

    ret = lwip_send(socket_fd, ReceiveBuff, sizeof(ReceiveBuff)-1, 0);
    if(ret < 0)
    {
        lwip_close(socket_fd);
        break;
    }
```

TCP
Server ip:192.168.1.22
Port :1234

Gateway Board



Port1~port4

MediaConverter_TJA1100

RJ45 <----> 100BASE-T1

04.

Hands-on – ENET1

Hands-on – ENET1: Objective

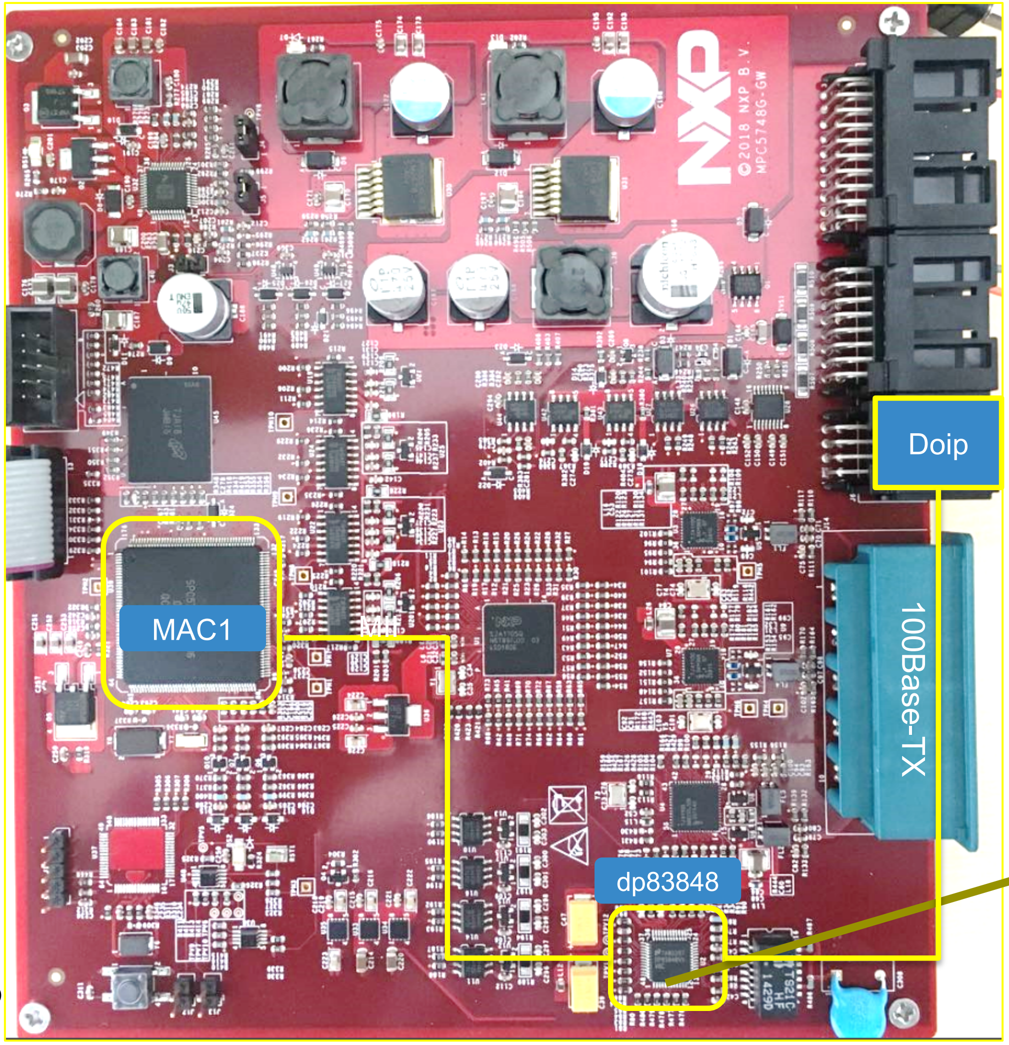
- ENET1 is configured similarly to ENET0.

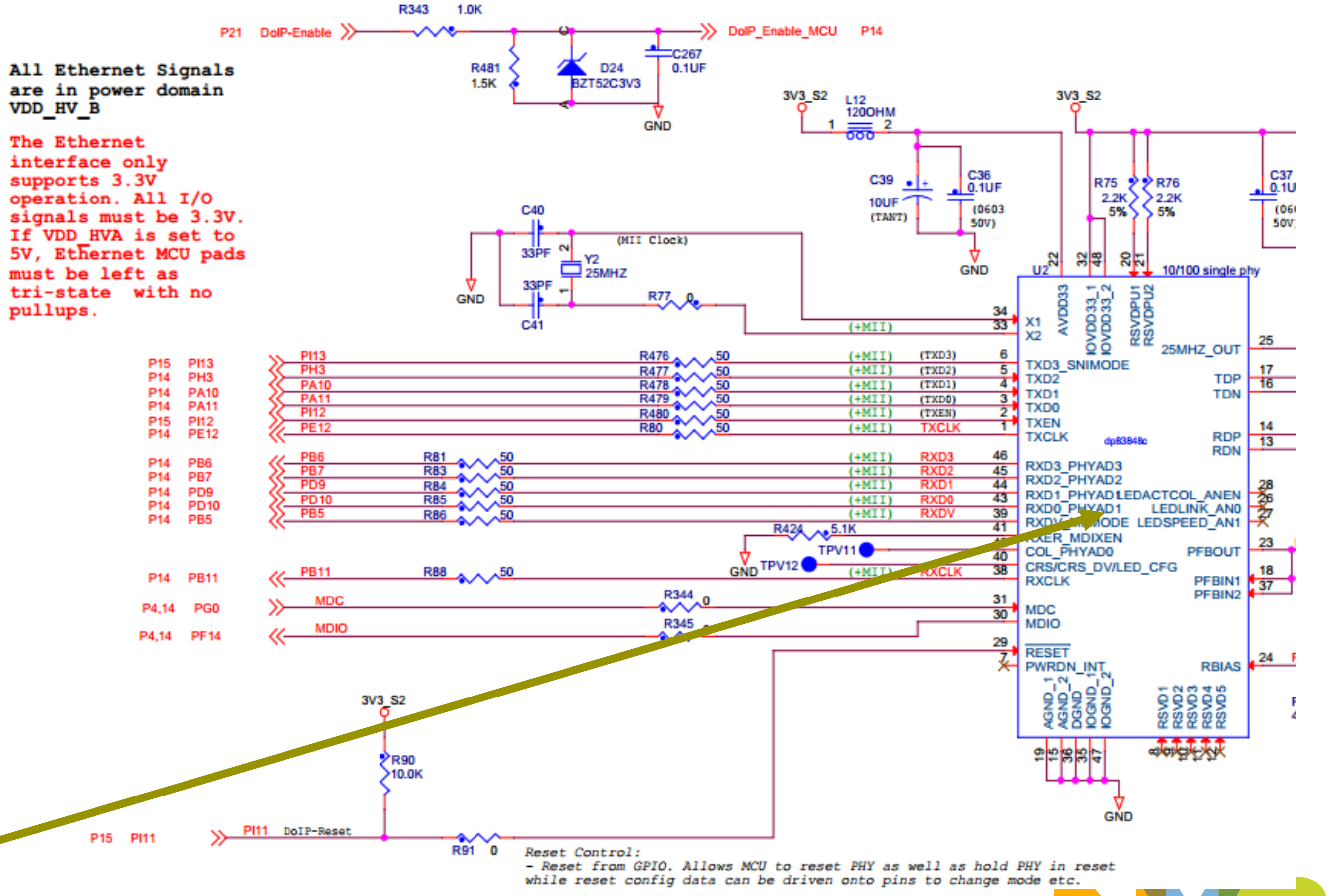
  - Unlike the previous section, the Phy(DP83848)for MCA1 directly connected.

- Please refer to section 3 if there is anything unclear

Hands-on – ENET1: Resources

- Resources to be used:
  - on-board user ENET1 ports





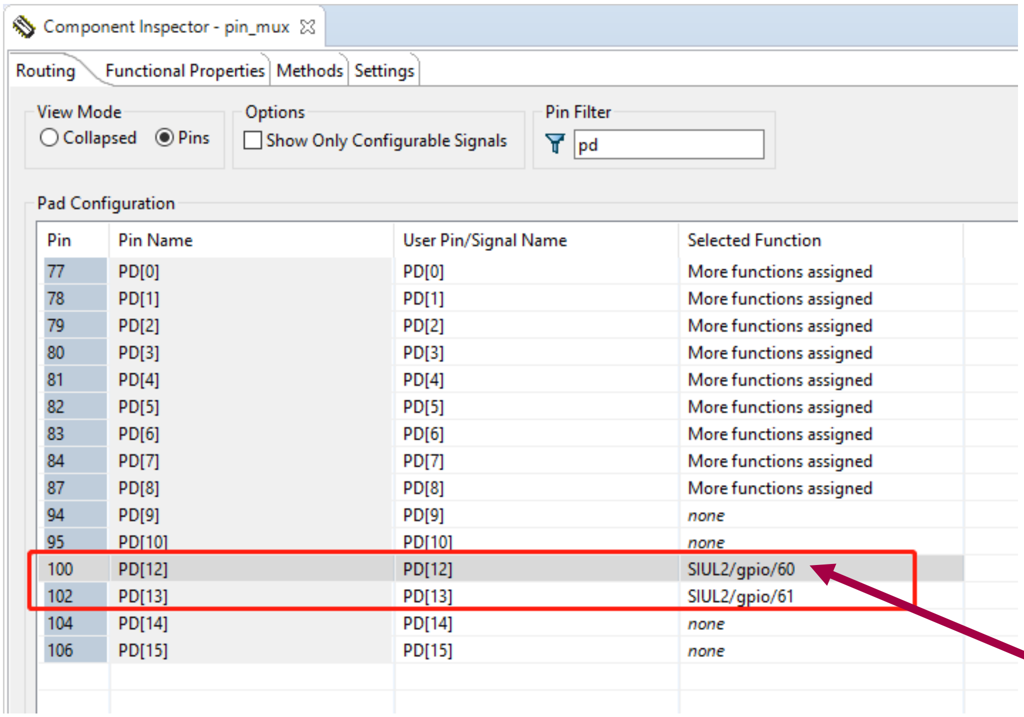
All Ethernet Signals are in power domain VDD_HV_B

The Ethernet interface only supports 3.3V operation. All I/O signals must be 3.3V. If VDD_HVA is set to 5V, Ethernet MCU pads must be left as tri-state with no pullups.

Reset Control:
- Reset from GPIO. Allows MCU to reset PHY as well as hold PHY in reset while reset config data can be driven onto pins to change mode etc.

Hands-on – ENET1: Modify-Peripheral Power Supply

- Enable the peripheral power supply
- Open ‘pin_mux’ component in ‘Component Inspector’ to configure pin routing
- SIUL2 tab -> GPIO 60 （61） > select the pin (one option) + direction output



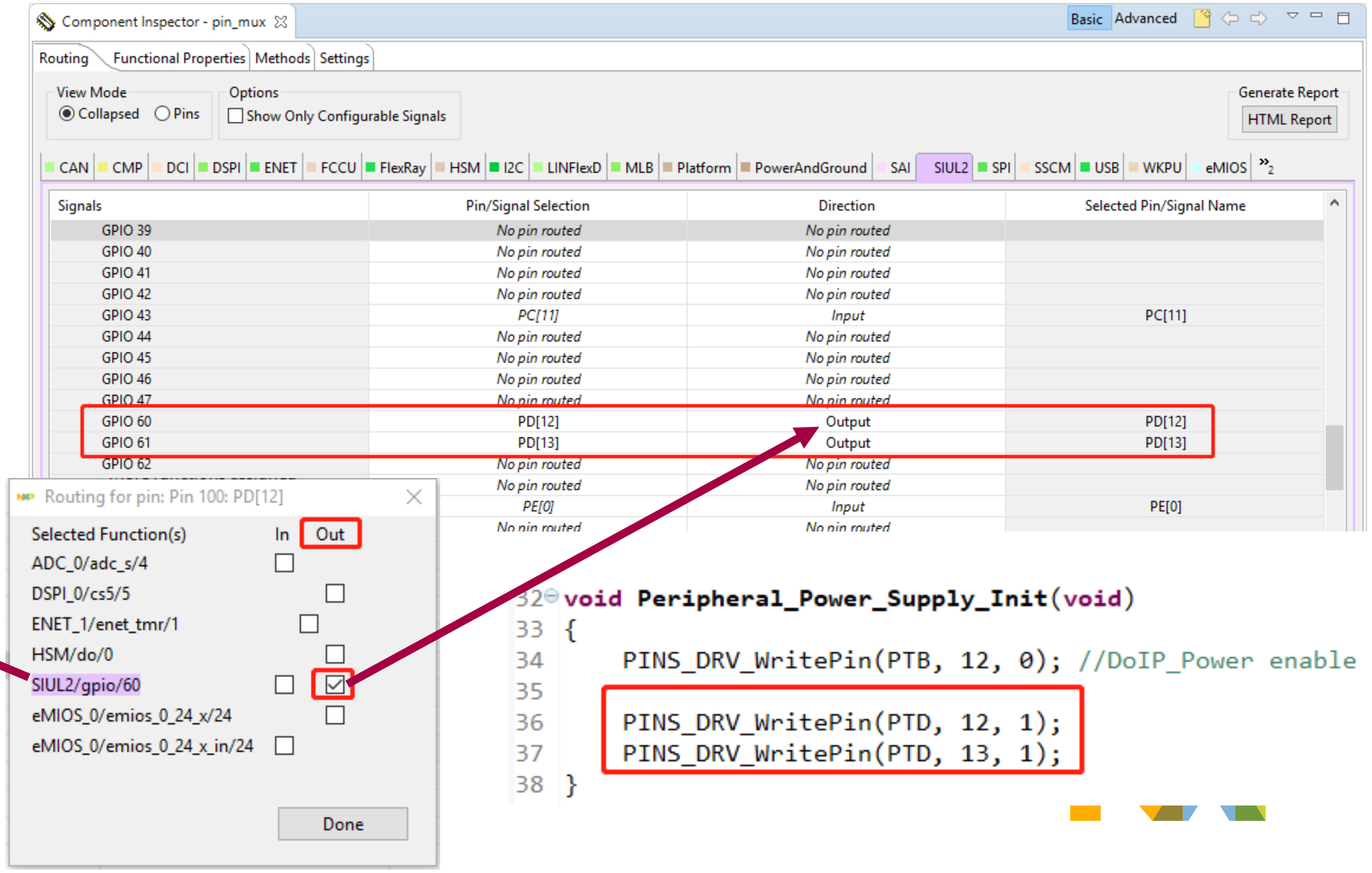
Component Inspector - pin_mux

Routing | Functional Properties | Methods | Settings

View Mode: ○ Collapsed [✓ ● Pins]
Options: ☐ Show Only Configurable Signals
Pin Filter: pd

Pad Configuration

Pin	Pin Name	User Pin/Signal Name	Selected Function
77	PD[0]	PD[0]	More functions assigned
78	PD[1]	PD[1]	More functions assigned
79	PD[2]	PD[2]	More functions assigned
80	PD[3]	PD[3]	More functions assigned
81	PD[4]	PD[4]	More functions assigned
82	PD[5]	PD[5]	More functions assigned
83	PD[6]	PD[6]	More functions assigned
84	PD[7]	PD[7]	More functions assigned
87	PD[8]	PD[8]	More functions assigned
94	PD[9]	PD[9]	*none*
95	PD[10]	PD[10]	*none*
100	PD[12]	PD[12]	SIUL2/gpio/60
102	PD[13]	PD[13]	SIUL2/gpio/61
104	PD[14]	PD[14]	*none*
106	PD[15]	PD[15]	*none*



Component Inspector - pin_mux

Routing | Functional Properties | Methods | Settings

View Mode: [✓ ● Collapsed] ○ Pins
Options: ☐ Show Only Configurable Signals
Generate Report: HTML Report

CAN | CMP | DCI | DSPI | ENET | FCCU | FlexRay | HSM | I2C | LINFlexD | MLB | Platform | PowerAndGround | SAI | SIUL2 | SPI | SSCM | USB | WKPU | eMIOS

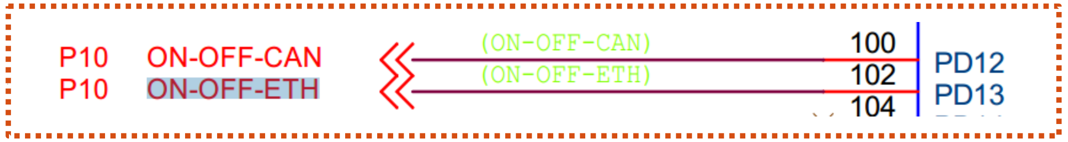
Signals	Pin/Signal Selection	Direction	Selected Pin/Signal Name
GPIO 39	*No pin routed*	*No pin routed*	
GPIO 40	*No pin routed*	*No pin routed*	
GPIO 41	*No pin routed*	*No pin routed*	
GPIO 42	*No pin routed*	*No pin routed*	
GPIO 43	*PC[11]*	*Input*	PC[11]
GPIO 44	*No pin routed*	*No pin routed*	
GPIO 45	*No pin routed*	*No pin routed*	
GPIO 46	*No pin routed*	*No pin routed*	
GPIO 47	*No pin routed*	*No pin routed*	
GPIO 60	PD[12]	Output	PD[12]
GPIO 61	PD[13]	Output	PD[13]
GPIO 62	*No pin routed*	*No pin routed*	
	No pin routed	*No pin routed*	
	PE[0]	*Input*	PE[0]
	No pin routed	*No pin routed*	

Routing for pin: Pin 100: PD[12]

Selected Function(s)	In	Out
ADC_0/adc_s/4	☐	
DSPI_0/cs5/5		☐
ENET_1/enet_tmr/1	☐	
HSM/do/0		☐
SIUL2/gpio/60	☐	☑
eMIOS_0/emios_0_24_x/24		☐
eMIOS_0/emios_0_24_x_in/24	☐	

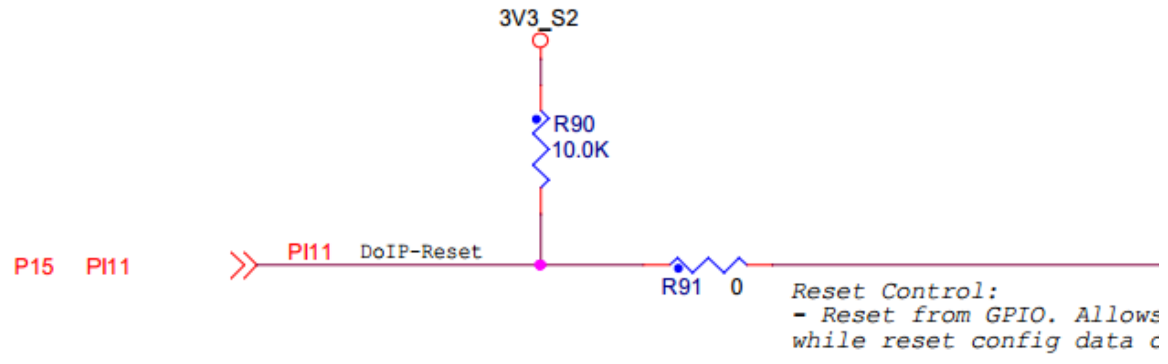
Done

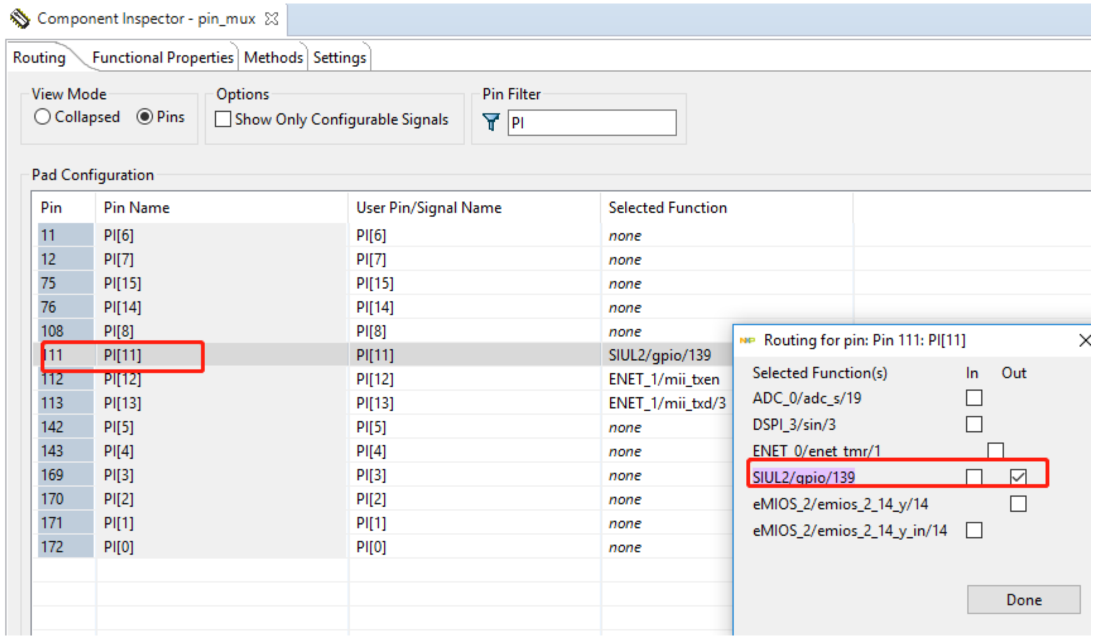
```
void Peripheral_Power_Supply_Init(void)
{
    PINS_DRV_WritePin(PTB, 12, 0); //DoIP_Power enable

    PINS_DRV_WritePin(PTD, 12, 1);
    PINS_DRV_WritePin(PTD, 13, 1);
}
```



Hands-on – ENET1: DP83848 configuration

- Initialize the DoIP-Reset pin
  - Initialize to high level





View Mode: ( ) Collapsed [✓ Pins]

Options: ☐ Show Only Configurable Signals

Pin Filter: PI

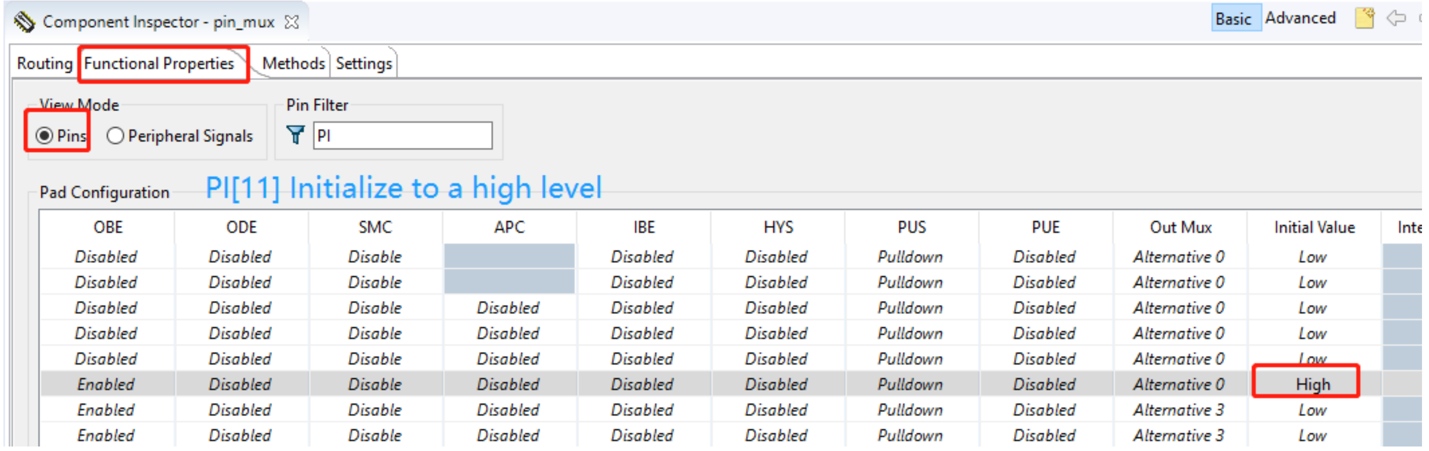
Pad Configuration

Pin	Pin Name	User Pin/Signal Name	Selected Function
11	PI[6]	PI[6]	*none*
12	PI[7]	PI[7]	*none*
75	PI[15]	PI[15]	*none*
76	PI[14]	PI[14]	*none*
108	PI[8]	PI[8]	*none*
111	PI[11]	PI[11]	SIUL2/gpio/139
112	PI[12]	PI[12]	ENET_1/mii_txen
113	PI[13]	PI[13]	ENET_1/mii_txd/3
142	PI[5]	PI[5]	*none*
143	PI[4]	PI[4]	*none*
169	PI[3]	PI[3]	*none*
170	PI[2]	PI[2]	*none*
171	PI[1]	PI[1]	*none*
172	PI[0]	PI[0]	*none*

Routing for pin: Pin 111: PI[11]

Selected Function(s)	In	Out
ADC_0/adc_s/19	☐	
DSPI_3/sin/3	☐	
ENET_0/enet_tmr/1	☐	
SIUL2/gpio/139	☐	✓
eMIOS_2/emios_2_14_y/14		☐
eMIOS_2/emios_2_14_y_in/14	☐	

Done



View Mode: [✓ Pins] ( ) Peripheral Signals

Pin Filter: PI

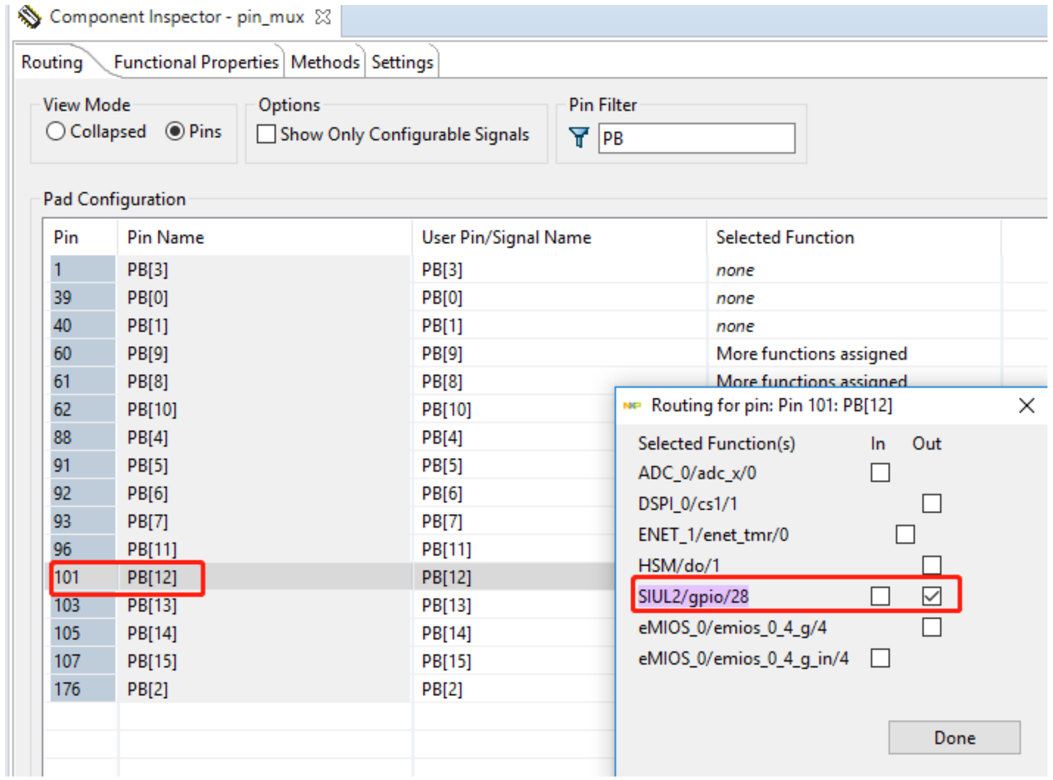
Pad Configuration

PI[11] Initialize to a high level

OBE	ODE	SMC	APC	IBE	HYS	PUS	PUE	Out Mux	Initial Value	Inte
Disabled	*Disabled*	*Disable*		*Disabled*	*Disabled*	*Pulldown*	*Disabled*	*Alternative 0*	*Low*	
Disabled	*Disabled*	*Disable*		*Disabled*	*Disabled*	*Pulldown*	*Disabled*	*Alternative 0*	*Low*	
Disabled	*Disabled*	*Disable*	*Disabled*	*Disabled*	*Disabled*	*Pulldown*	*Disabled*	*Alternative 0*	*Low*	
Disabled	*Disabled*	*Disable*	*Disabled*	*Disabled*	*Disabled*	*Pulldown*	*Disabled*	*Alternative 0*	*Low*	
Disabled	*Disabled*	*Disable*	*Disabled*	*Disabled*	*Disabled*	*Pulldown*	*Disabled*	*Alternative 0*	*Low*	
Enabled	*Disabled*	*Disable*	*Disabled*	*Disabled*	*Disabled*	*Pulldown*	*Disabled*	*Alternative 0*	High	
Enabled	*Disabled*	*Disable*	*Disabled*	*Disabled*	*Disabled*	*Pulldown*	*Disabled*	*Alternative 3*	*Low*	
Enabled	*Disabled*	*Disable*	*Disabled*	*Disabled*	*Disabled*	*Pulldown*	*Disabled*	*Alternative 3*	*Low*	

Hands-on – ENET1: DP83848 configuration

- Enable the DoIP Phy`s Power



Component Inspector - pin_mux

Routing | Functional Properties | Methods | Settings

View Mode: ( ) Collapsed [✓ Pins]

Options: ☐ Show Only Configurable Signals

Pin Filter: PB

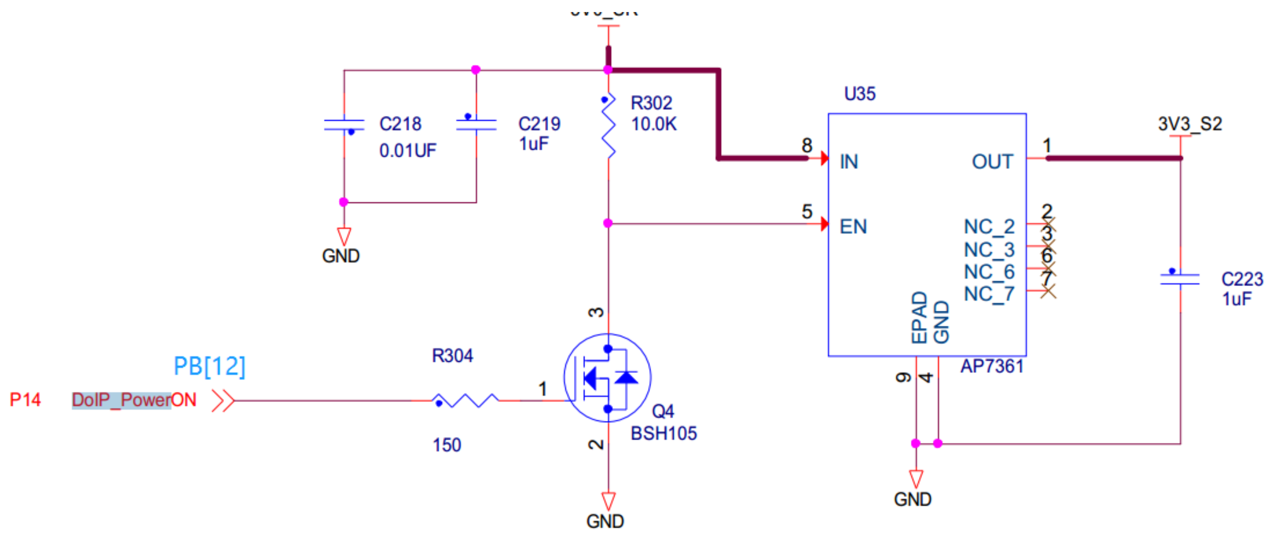
Pad Configuration

Pin	Pin Name	User Pin/Signal Name	Selected Function
1	PB[3]	PB[3]	*none*
39	PB[0]	PB[0]	*none*
40	PB[1]	PB[1]	*none*
60	PB[9]	PB[9]	More functions assigned
61	PB[8]	PB[8]	More functions assigned
62	PB[10]	PB[10]	
88	PB[4]	PB[4]	
91	PB[5]	PB[5]	
92	PB[6]	PB[6]	
93	PB[7]	PB[7]	
96	PB[11]	PB[11]	
101	PB[12]	PB[12]	
103	PB[13]	PB[13]	
105	PB[14]	PB[14]	
107	PB[15]	PB[15]	
176	PB[2]	PB[2]	

Routing for pin: Pin 101: PB[12]

Selected Function(s)	In	Out
ADC_0/adc_x/0	☐	
DSPI_0/cs1/1		☐
ENET_1/enet_tmr/0	☐	
HSM/do/1		☐
SIUL2/gpio/28	☐	☑
eMIOS_0/emios_0_4_g/4		☐
eMIOS_0/emios_0_4_g_in/4	☐	

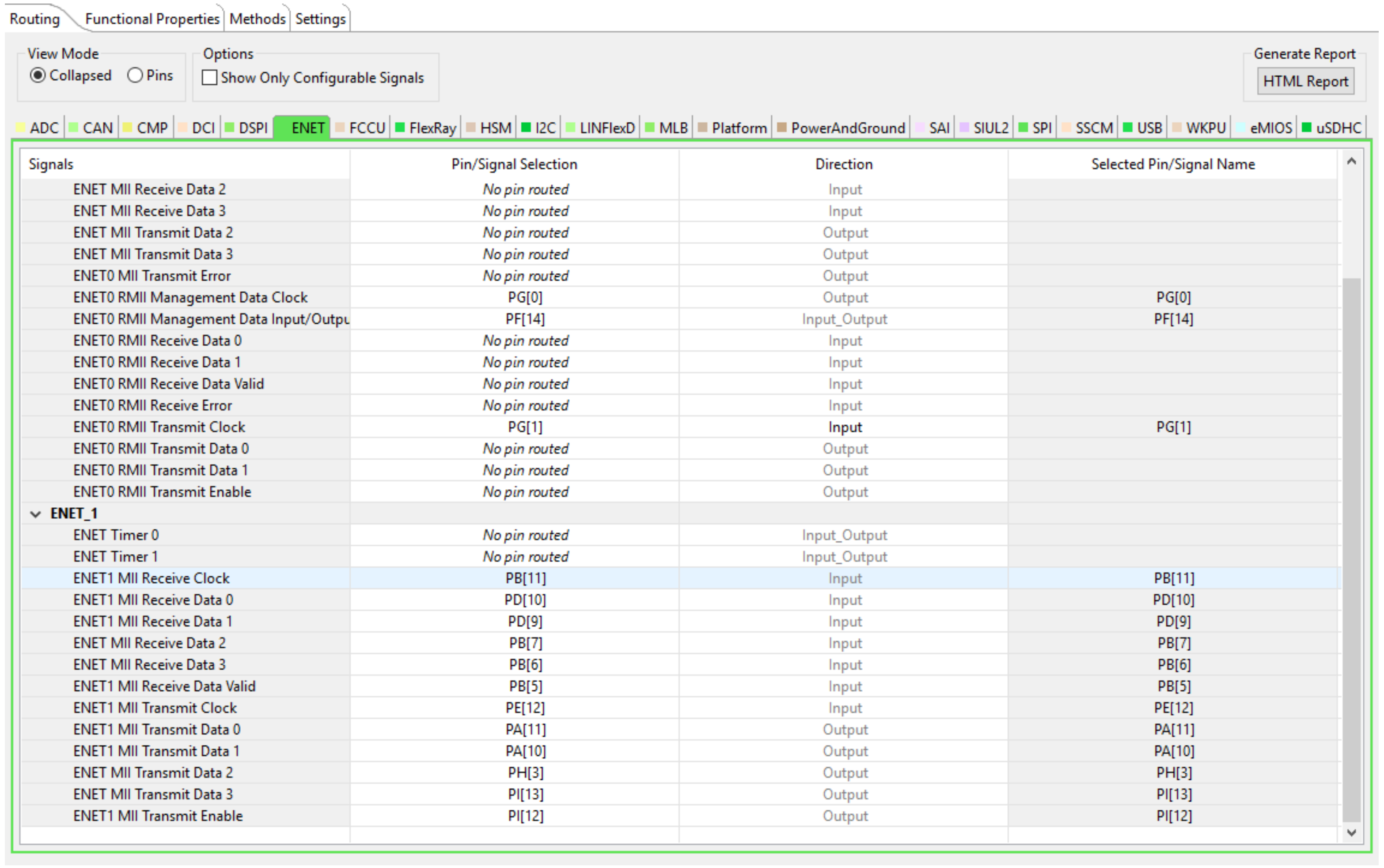
Done



```
void Peripheral_Power_Supply_Init(void)
{
    PINS_DRV_WritePin(PTB, 12, 0); //DoIP_Power enable

    PINS_DRV_WritePin(PTD, 12, 1);
    PINS_DRV_WritePin(PTD, 13, 1);
}
```

Hands-on – ENET1: ENET configuration

- In ‘pin_mux’ component –> Routing(Collapsed)– select ‘ENET’
  - -Configuration the ports according to the schematic.

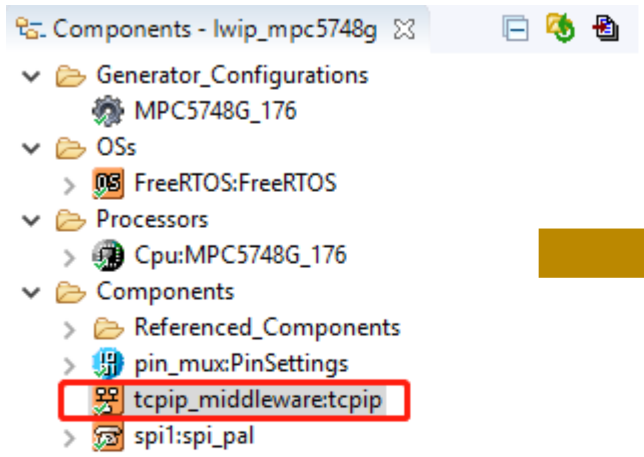

Routing
Functional Properties
Methods
Settings
Generate Report
HTML Report
ADC | CAN | CMP | DCI | DSPI | ENET | FCCU | FlexRay | HSM | I2C | LINFlexD | MLB | Platform | PowerAndGround | SAI | SIUL2 | SPI | SSCM | USB | WKPU | eMIOS | uSDHC

View Mode: [✓ Collapsed] Pins

Options: ☐ Show Only Configurable Signals

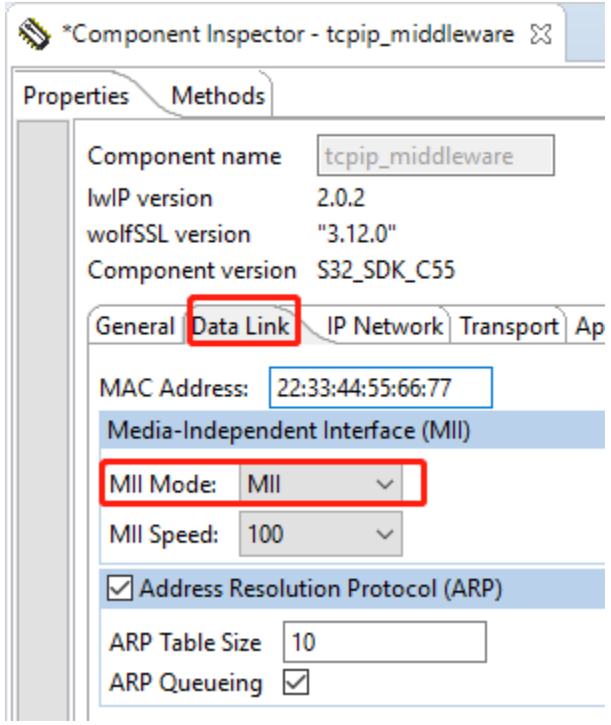
Signals	Pin/Signal Selection	Direction	Selected Pin/Signal Name
ENET MII Receive Data 2	*No pin routed*	Input	
ENET MII Receive Data 3	*No pin routed*	Input	
ENET MII Transmit Data 2	*No pin routed*	Output	
ENET MII Transmit Data 3	*No pin routed*	Output	
ENET0 MII Transmit Error	*No pin routed*	Output	
ENET0 RMII Management Data Clock	PG[0]	Output	PG[0]
ENET0 RMII Management Data Input/Outpu	PF[14]	Input_Output	PF[14]
ENET0 RMII Receive Data 0	*No pin routed*	Input	
ENET0 RMII Receive Data 1	*No pin routed*	Input	
ENET0 RMII Receive Data Valid	*No pin routed*	Input	
ENET0 RMII Receive Error	*No pin routed*	Input	
ENET0 RMII Transmit Clock	PG[1]	Input	PG[1]
ENET0 RMII Transmit Data 0	*No pin routed*	Output	
ENET0 RMII Transmit Data 1	*No pin routed*	Output	
ENET0 RMII Transmit Enable	*No pin routed*	Output	
ENET_1			
ENET Timer 0	*No pin routed*	Input_Output	
ENET Timer 1	*No pin routed*	Input_Output	
ENET1 MII Receive Clock	PB[11]	Input	PB[11]
ENET1 MII Receive Data 0	PD[10]	Input	PD[10]
ENET1 MII Receive Data 1	PD[9]	Input	PD[9]
ENET MII Receive Data 2	PB[7]	Input	PB[7]
ENET MII Receive Data 3	PB[6]	Input	PB[6]
ENET1 MII Receive Data Valid	PB[5]	Input	PB[5]
ENET1 MII Transmit Clock	PE[12]	Input	PE[12]
ENET1 MII Transmit Data 0	PA[11]	Output	PA[11]
ENET1 MII Transmit Data 1	PA[10]	Output	PA[10]
ENET MII Transmit Data 2	PH[3]	Output	PH[3]
ENET MII Transmit Data 3	PI[13]	Output	PI[13]
ENET1 MII Transmit Enable	PI[12]	Output	PI[12]

Hands-on – ENET1: lwip middleware configuration



Components - lwip_mpc5748g

- Generator_Configurations
  - MPC5748G_176
- OSs
  - FreeRTOS:FreeRTOS
- Processors
  - Cpu:MPC5748G_176
- Components
  - Referenced_Components
  - pin_mux:PinSettings
  - tcpip_middleware:tcpip
  - spi1:spi_pal



*Component Inspector - tcpip_middleware

Properties | Methods

Component name	tcpip_middleware
lwIP version	2.0.2
wolfSSL version	"3.12.0"
Component version	S32_SDK_C55

General | Data Link | IP Network | Transport | Ap

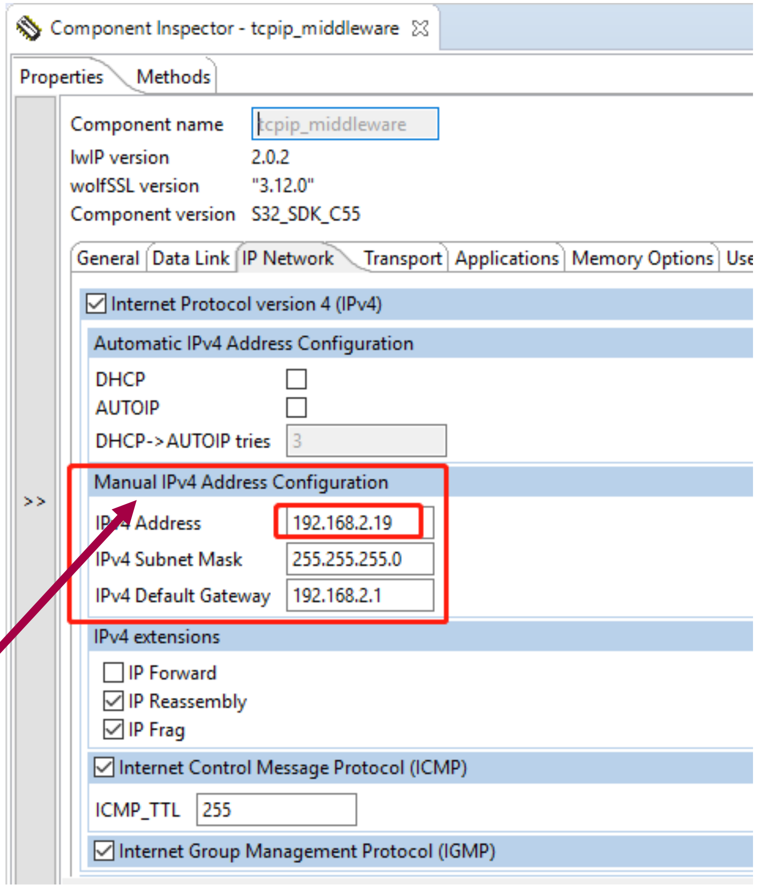
MAC Address: 22:33:44:55:66:77

Media-Independent Interface (MII)

MII Mode:	MII
MII Speed:	100

☑ Address Resolution Protocol (ARP)

ARP Table Size	10
ARP Queueing	☑



Component Inspector - tcpip_middleware

Properties | Methods

Component name	tcpip_middleware
lwIP version	2.0.2
wolfSSL version	"3.12.0"
Component version	S32_SDK_C55

General | Data Link | IP Network | Transport | Applications | Memory Options | Use

☑ Internet Protocol version 4 (IPv4)

Automatic IPv4 Address Configuration

DHCP	☐
AUTOIP	☐
DHCP->AUTOIP tries	3

Manual IPv4 Address Configuration

IPv4 Address	192.168.2.19
IPv4 Subnet Mask	255.255.255.0
IPv4 Default Gateway	192.168.2.1

IPv4 extensions

☐ IP Forward
☑ IP Reassembly
☑ IP Frag

☑ Internet Control Message Protocol (ICMP)

ICMP_TTL 255

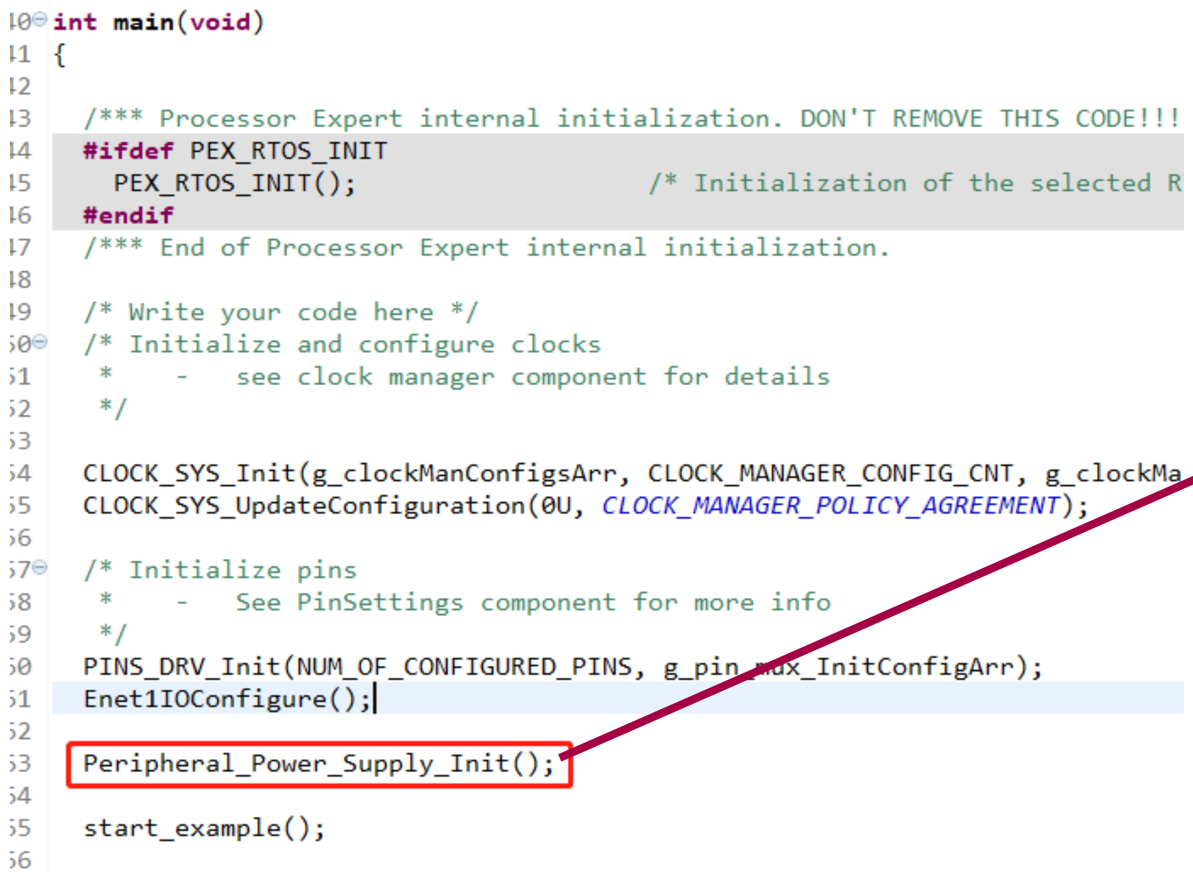
☑ Internet Group Management Protocol (IGMP)

Define it by yourself

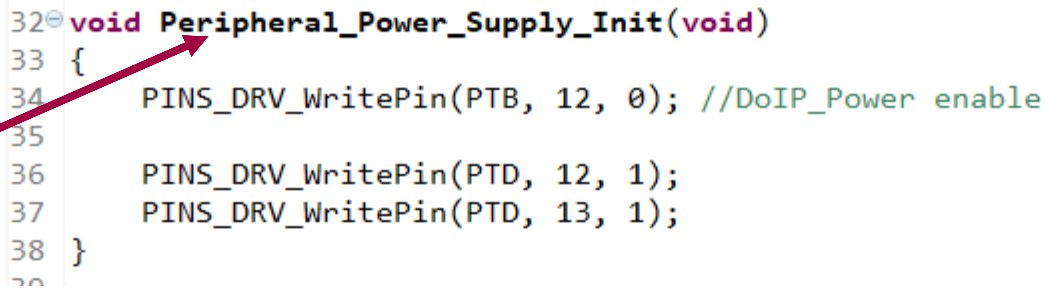
Hands-on – ENET1: Application Code

(1) **Peripheral Power Supply:**

- {Project Name} -> Source -> main.c



```
int main(void)
{

  /*** Processor Expert internal initialization. DON'T REMOVE THIS CODE!!!
  #ifdef PEX_RTOS_INIT
    PEX_RTOS_INIT();                    /* Initialization of the selected R
  #endif
  /*** End of Processor Expert internal initialization.

  /* Write your code here */
  /* Initialize and configure clocks
   *    -   see clock manager component for details
   */

  CLOCK_SYS_Init(g_clockManConfigsArr, CLOCK_MANAGER_CONFIG_CNT, g_clockMa
  CLOCK_SYS_UpdateConfiguration(0U, CLOCK_MANAGER_POLICY_AGREEMENT);

  /* Initialize pins
   *    -   See PinSettings component for more info
   */
  PINS_DRV_Init(NUM_OF_CONFIGURED_PINS, g_pin_mux_InitConfigArr);
  Enet1IOConfigure();

  Peripheral_Power_Supply_Init();

  start_example();

```



```
void Peripheral_Power_Supply_Init(void)
{
    PINS_DRV_WritePin(PTB, 12, 0); //DoIP_Power enable

    PINS_DRV_WritePin(PTD, 12, 1);
    PINS_DRV_WritePin(PTD, 13, 1);
}
```

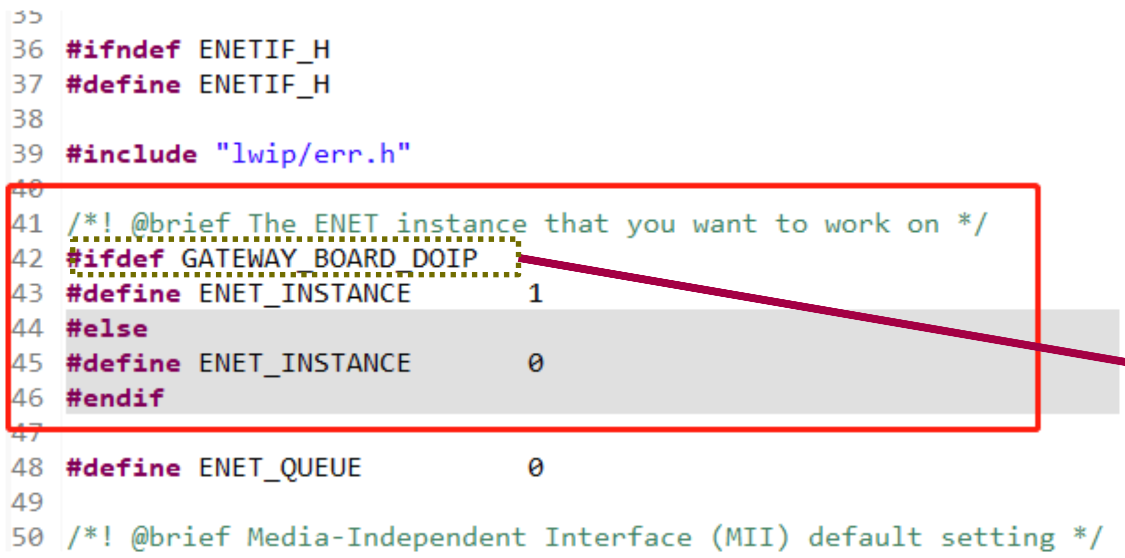
Hands-on – ENET1: Application Code

(2) Fix a problem that exists in our SDK

- {Project Name} -> SDK -> middleware -> tcpip -> tcpip_stack -> ports -> netif -> enetif -> enetif.h

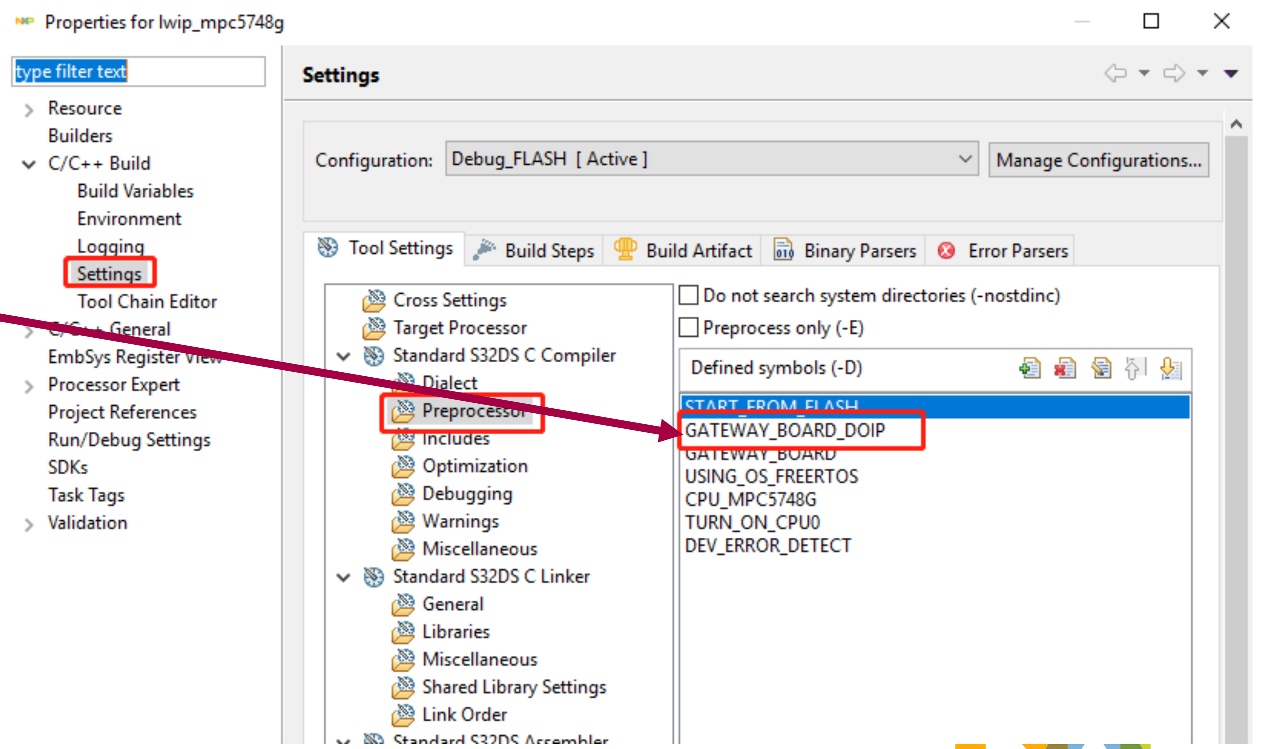
Note:

If you have a patched after update the SDK, you can ignore this modification



```
#ifndef ENETIF_H
#define ENETIF_H

#include "lwip/err.h"

/*! @brief The ENET instance that you want to work on */
#ifdef GATEWAY_BOARD_DOIP
#define ENET_INSTANCE        1
#else
#define ENET_INSTANCE        0
#endif

#define ENET_QUEUE           0

/*! @brief Media-Independent Interface (MII) default setting */
```

Add macro definition



Properties for lwip_mpc5748g

Settings

Configuration: Debug_FLASH [Active]

Tool Settings → Standard S32DS C Compiler → Preprocessor

- [ ] Do not search system directories (-nostdinc)
- [ ] Preprocess only (-E)

Defined symbols (-D)

```
START_FROM_FLASH
GATEWAY_BOARD_DOIP
GATEWAY_BOARD
USING_OS_FREERTOS
CPU_MPC5748G
TURN_ON_CPU0
DEV_ERROR_DETECT
```

Explanation:

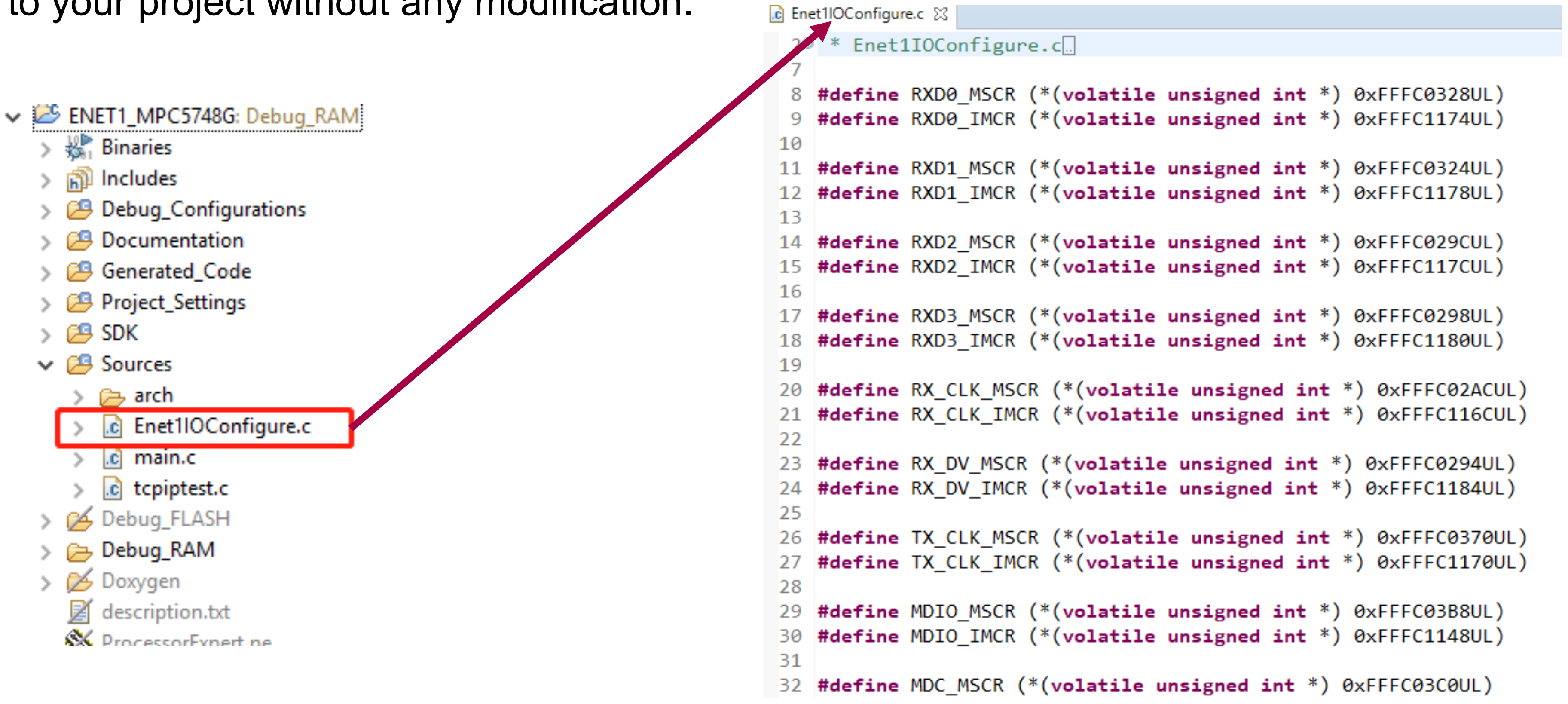
The SDK only implements the configuration for MAC0, and the code shown above needs to be added to support MAC1

Hands-on – ENET1: Application Code

(3) Fix a problem that exists in our PE

CASE: There is a problem with the generated code for ENET1`s GPIO configuration.

You can refer to the sample demo(The project of *ENET1_MPC5748G*) we provided. And copy the *Enet1IOConfigure.c* file to your project without any modification.



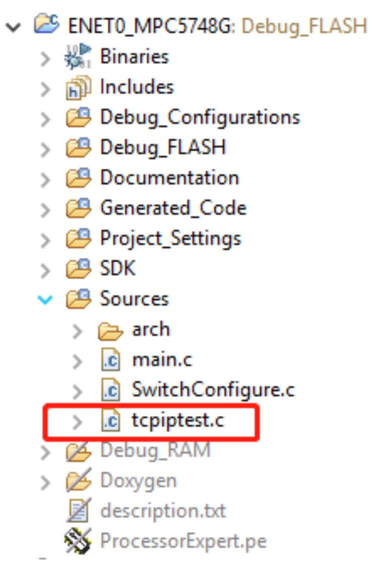
```c
 * Enet1IOConfigure.c

#define RXD0_MSCR (*(volatile unsigned int *) 0xFFFC0328UL)
#define RXD0_IMCR (*(volatile unsigned int *) 0xFFFC1174UL)

#define RXD1_MSCR (*(volatile unsigned int *) 0xFFFC0324UL)
#define RXD1_IMCR (*(volatile unsigned int *) 0xFFFC1178UL)

#define RXD2_MSCR (*(volatile unsigned int *) 0xFFFC029CUL)
#define RXD2_IMCR (*(volatile unsigned int *) 0xFFFC117CUL)

#define RXD3_MSCR (*(volatile unsigned int *) 0xFFFC0298UL)
#define RXD3_IMCR (*(volatile unsigned int *) 0xFFFC1180UL)

#define RX_CLK_MSCR (*(volatile unsigned int *) 0xFFFC02ACUL)
#define RX_CLK_IMCR (*(volatile unsigned int *) 0xFFFC116CUL)

#define RX_DV_MSCR (*(volatile unsigned int *) 0xFFFC0294UL)
#define RX_DV_IMCR (*(volatile unsigned int *) 0xFFFC1184UL)

#define TX_CLK_MSCR (*(volatile unsigned int *) 0xFFFC0370UL)
#define TX_CLK_IMCR (*(volatile unsigned int *) 0xFFFC1170UL)

#define MDIO_MSCR (*(volatile unsigned int *) 0xFFFC03B8UL)
#define MDIO_IMCR (*(volatile unsigned int *) 0xFFFC1148UL)

#define MDC_MSCR (*(volatile unsigned int *) 0xFFFC03C0UL)
```

Hands-on – ENET1: Application Code

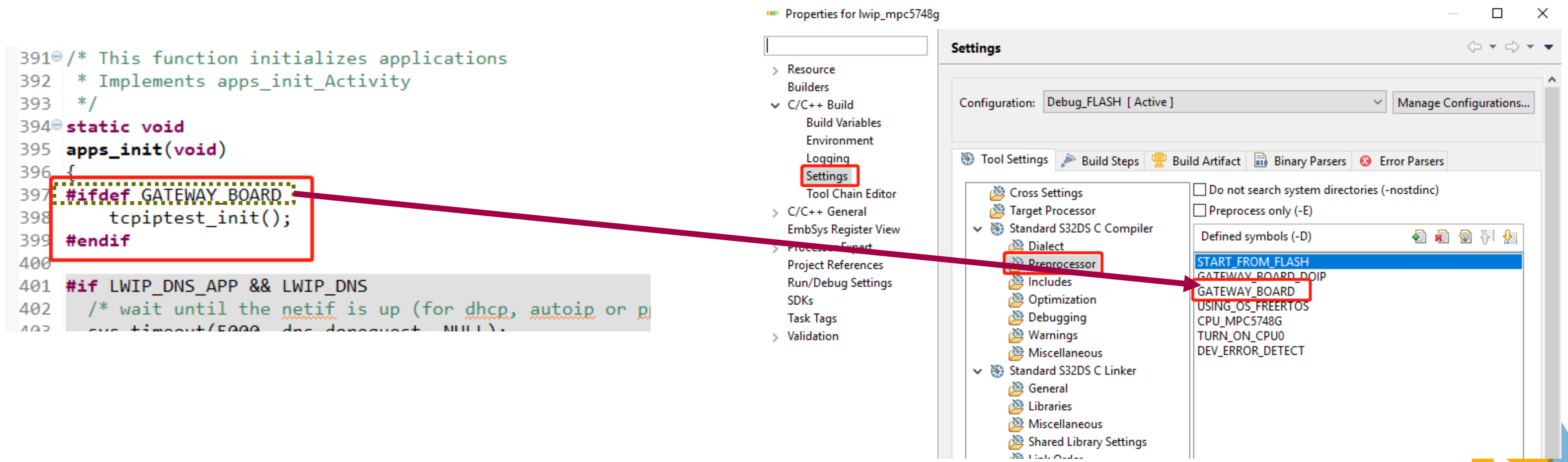
(4) Add client application for lwip_tcp

- You can refer to the sample demo(The project of *ENET0_MPC5748G*) we provided. And copy the *tcpiptest.c* file to your project without any modification.

- Add clicent application call function under the test.c file
  - {Project Name} -> SDK -> middleware -> tcpip -> tcpip_stack -> demo -> test.c



```
ENET0_MPC5748G: Debug_FLASH
  Binaries
  Includes
  Debug_Configurations
  Debug_FLASH
  Documentation
  Generated_Code
  Project_Settings
  SDK
  Sources
    arch
    main.c
    SwitchConfigure.c
    tcpiptest.c
  Debug_RAM
  Doxygen
  description.txt
  ProcessorExpert.pe
```



```
/* This function initializes applications
 * Implements apps_init_Activity
 */
static void
apps_init(void)
{
#ifdef GATEWAY_BOARD
    tcpiptest_init();
#endif

#if LWIP_DNS_APP && LWIP_DNS
  /* wait until the netif is up (for dhcp, autoip or p
```

Properties for lwip_mpc5748g

Settings

Configuration: Debug_FLASH [Active]

☐ Do not search system directories (-nostdinc)

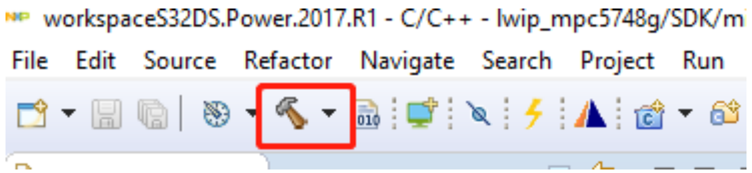
☐ Preprocess only (-E)

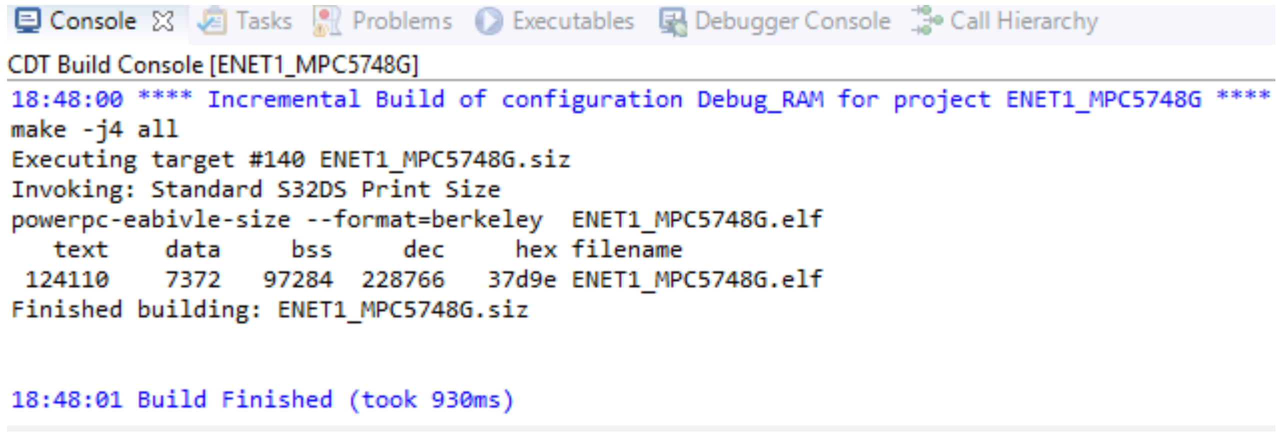
Defined symbols (-D)

```
START_FROM_FLASH
GATEWAY_BOARD_DOIP
GATEWAY_BOARD
USING_OS_FREERTOS
CPU_MPC5748G
TURN_ON_CPU0
DEV_ERROR_DETECT
```

Hands-on – ENET1: Build and Debug

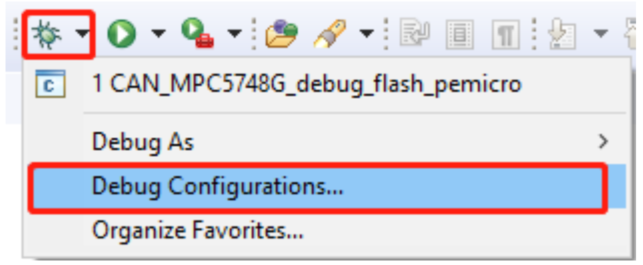
- Click the ‘build project’ button – make sure there are no compilation errors

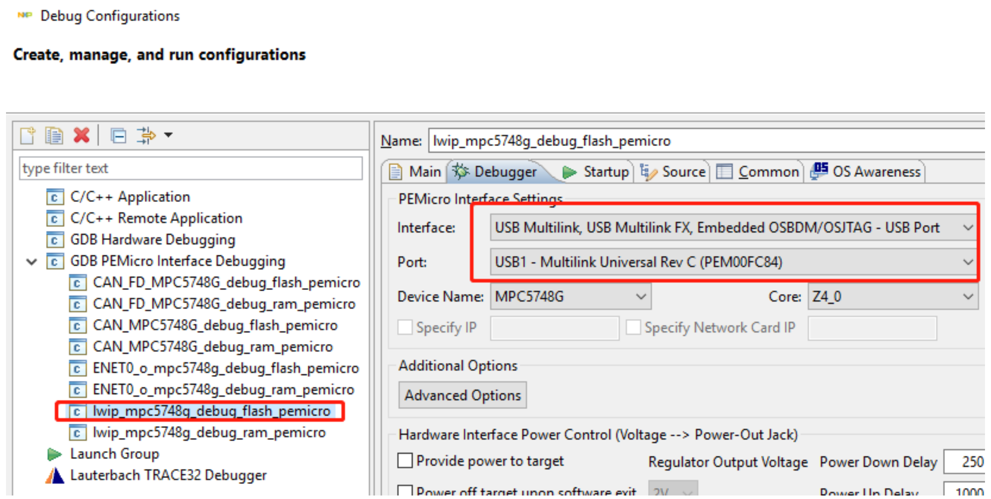

workspaceS32DS.Power.2017.R1 - C/C++ - lwip_mpc5748g/SDK/m
File Edit Source Refactor Navigate Search Project Run



```
CDT Build Console [ENET1_MPC5748G]
18:48:00 **** Incremental Build of configuration Debug_RAM for project ENET1_MPC5748G ****
make -j4 all
Executing target #140 ENET1_MPC5748G.siz
Invoking: Standard S32DS Print Size
powerpc-eabivle-size --format=berkeley  ENET1_MPC5748G.elf
   text    data     bss     dec     hex filename
 124110    7372   97284  228766   37d9e ENET1_MPC5748G.elf
Finished building: ENET1_MPC5748G.siz


18:48:01 Build Finished (took 930ms)
```

- Select the correct debug configuration and interface to debug the application




Debug Configurations

Create, manage, and run configurations

type filter text

- C/C++ Application
- C/C++ Remote Application
- GDB Hardware Debugging
- GDB PEMicro Interface Debugging
  - CAN_FD_MPC5748G_debug_flash_pemicro
  - CAN_FD_MPC5748G_debug_ram_pemicro
  - CAN_MPC5748G_debug_flash_pemicro
  - CAN_MPC5748G_debug_ram_pemicro
  - ENET0_o_mpc5748g_debug_flash_pemicro
  - ENET0_o_mpc5748g_debug_ram_pemicro
  - lwip_mpc5748g_debug_flash_pemicro
  - lwip_mpc5748g_debug_ram_pemicro
- Launch Group
- Lauterbach TRACE32 Debugger

Name: lwip_mpc5748g_debug_flash_pemicro

Main | Debugger | Startup | Source | Common | OS Awareness

PEMicro Interface Settings

Interface: USB Multilink, USB Multilink FX, Embedded OSBDM/OSJTAG - USB Port

Port: USB1 - Multilink Universal Rev C (PEM00FC84)

Device Name: MPC5748G   Core: Z4_0

☐ Specify IP   ☐ Specify Network Card IP

Additional Options

Advanced Options

Hardware Interface Power Control (Voltage --> Power-Out Jack)

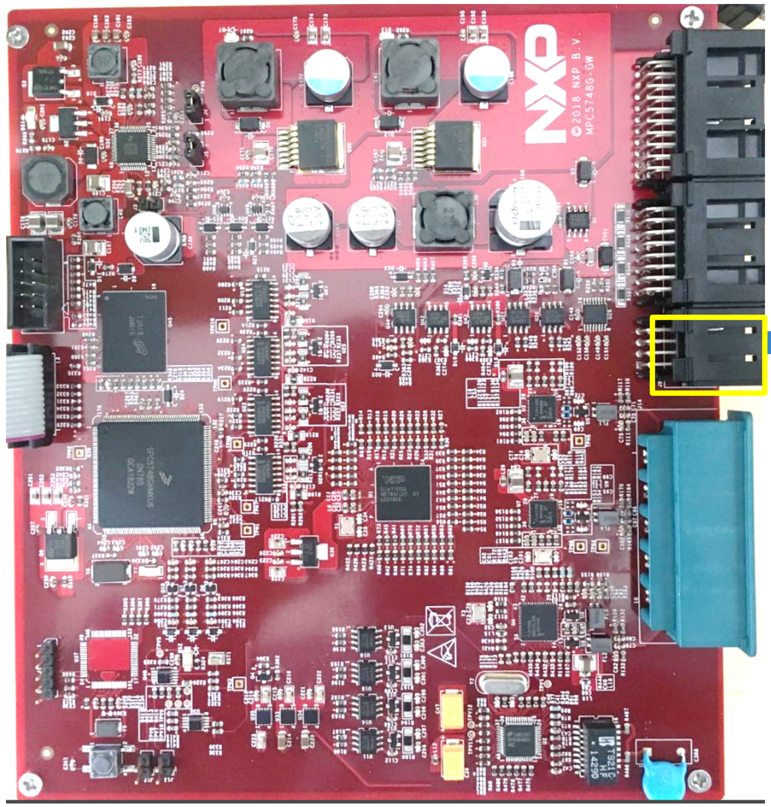
☐ Provide power to target   Regulator Output Voltage   Power Down Delay 250

☐ Power off target upon software exit   2V   Power Up Delay 1000

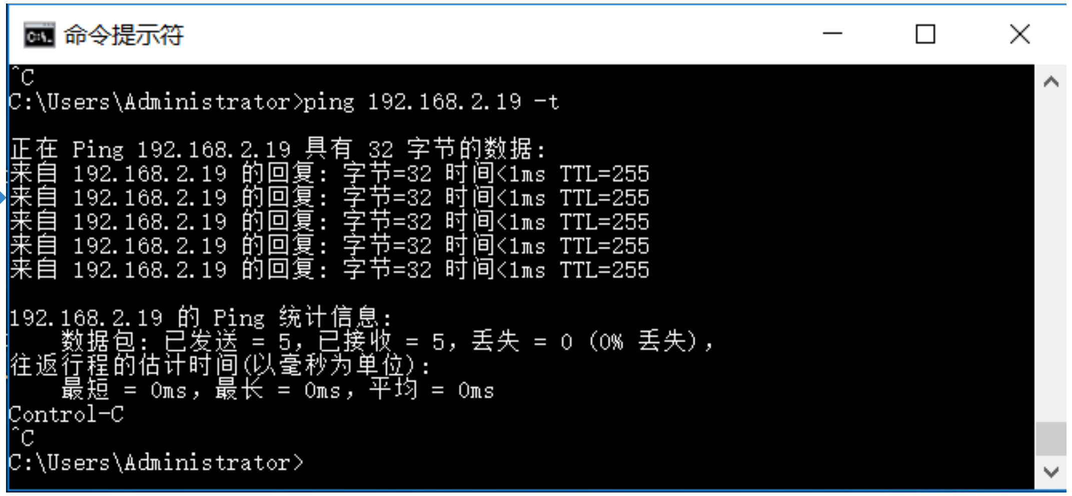
Hands-on – ENET1: Build and Debug

TEST: Ping the board from PC:

Gateway Board



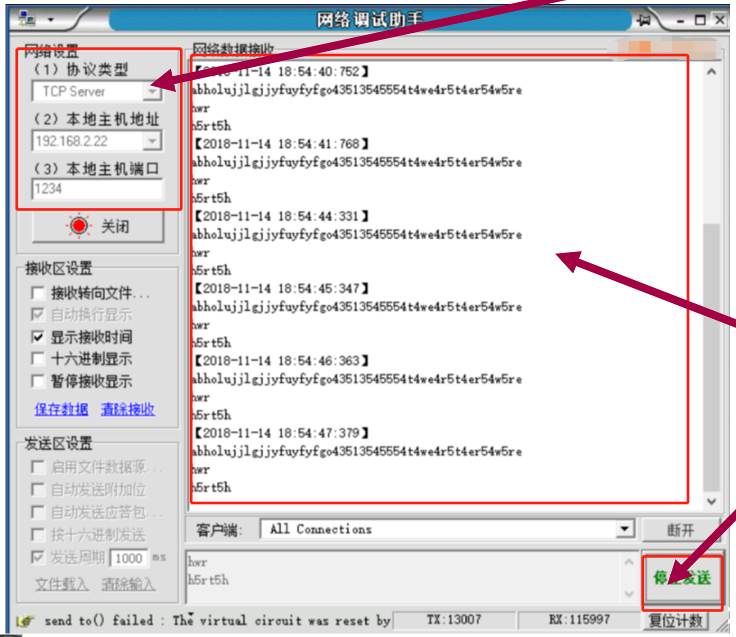
PC terminal



```
^C
C:\Users\Administrator>ping 192.168.2.19 -t

正在 Ping 192.168.2.19 具有 32 字节的数据:
来自 192.168.2.19 的回复: 字节=32 时间<1ms TTL=255
来自 192.168.2.19 的回复: 字节=32 时间<1ms TTL=255
来自 192.168.2.19 的回复: 字节=32 时间<1ms TTL=255
来自 192.168.2.19 的回复: 字节=32 时间<1ms TTL=255
来自 192.168.2.19 的回复: 字节=32 时间<1ms TTL=255

192.168.2.19 的 Ping 统计信息:
    数据包: 已发送 = 5，已接收 = 5，丢失 = 0 (0% 丢失)，
往返行程的估计时间(以毫秒为单位):
    最短 = 0ms，最长 = 0ms，平均 = 0ms
Control-C
^C
C:\Users\Administrator>
```

Hands-on – ENET1: Build and Debug

TEST: LWIP_tcp Client:



网络调试助手

网络设置
（1）协议类型: TCP Server
（2）本地主机地址: 192.168.2.22
（3）本地主机端口: 1234
关闭

接收区设置
- [ ] 接收转向文件...
- [x] 自动换行显示
- [x] 显示接收时间
- [ ] 十六进制显示
- [ ] 暂停接收显示

保存数据 清除接收

发送区设置
- [ ] 启用文件数据源...
- [ ] 自动发送附加位
- [ ] 自动发送应答包...
- [ ] 按十六进制发送
- [x] 发送周期 1000 ms

文件载入 清除输入

网络数据接收

```
【2018-11-14 18:54:40:752】
abholujjlgjjyfuyfyfgo43513545554t4we4r5t4er54w5re
hwr
h5rt5h
【2018-11-14 18:54:41:768】
abholujjlgjjyfuyfyfgo43513545554t4we4r5t4er54w5re
hwr
h5rt5h
【2018-11-14 18:54:44:331】
abholujjlgjjyfuyfyfgo43513545554t4we4r5t4er54w5re
hwr
h5rt5h
【2018-11-14 18:54:45:347】
abholujjlgjjyfuyfyfgo43513545554t4we4r5t4er54w5re
hwr
h5rt5h
【2018-11-14 18:54:46:363】
abholujjlgjjyfuyfyfgo43513545554t4we4r5t4er54w5re
hwr
h5rt5h
【2018-11-14 18:54:47:379】
abholujjlgjjyfuyfyfgo43513545554t4we4r5t4er54w5re
hwr
h5rt5h
```

客户端: All Connections 断开

```
hwr
h5rt5h
```

停止发送

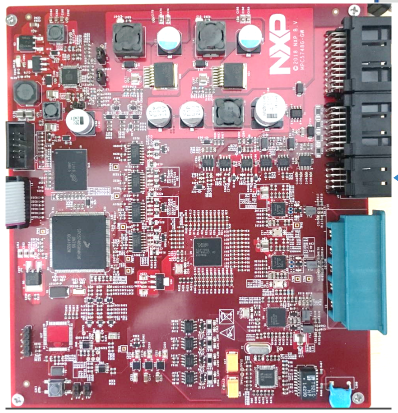
```
send to() failed : The virtual circuit was reset by    TX:13007    RX:115997
```

复位计数

```
memset(&server_addr, 0, sizeof(server_addr));
server_addr.sin_family = AF_INET;
server_addr.sin_addr.s_addr = inet_addr("192.168.2.22");
server_addr.sin_port =  PP_HTONS(PORT);
server_addr.sin_len = sizeof(server_addr);

ret= lwip_connect(socket_fd, (struct sockaddr*)&server_addr, sizeof(struct sockaddr));
if(ret != ERR_OK)
{
    vTaskDelete(NULL);
}

for(; ;)
{
    ret = lwip_read(socket_fd, ReceiveBuff, sizeof(ReceiveBuff));
    if(ret == -1)
    {
        lwip_close(socket_fd);
        break;
    }

    ret = lwip_send(socket_fd, ReceiveBuff, sizeof(ReceiveBuff)-1, 0);
    if(ret < 0)
    {
        lwip_close(socket_fd);
        break;
    }
```

TCP
Server ip:192.168.2.22
Port :1234

Gateway Board



05.

Hands-on – UART

Hands-on – UART: Objective

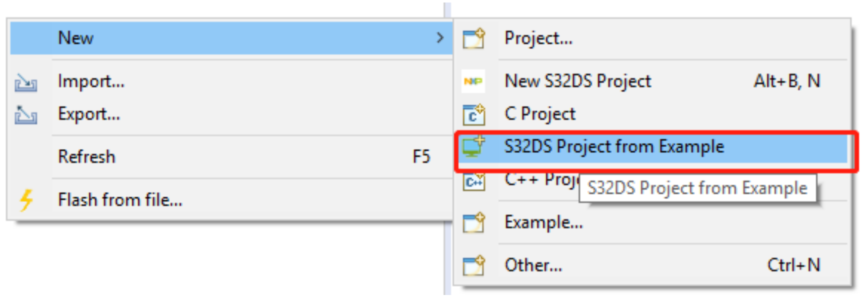
- Features of UART module on MPC5748G
- How to set a pin as output/input with SDK
- How to configure the port of UART
- How to modify an existing SDK project with S32DS to suit this board

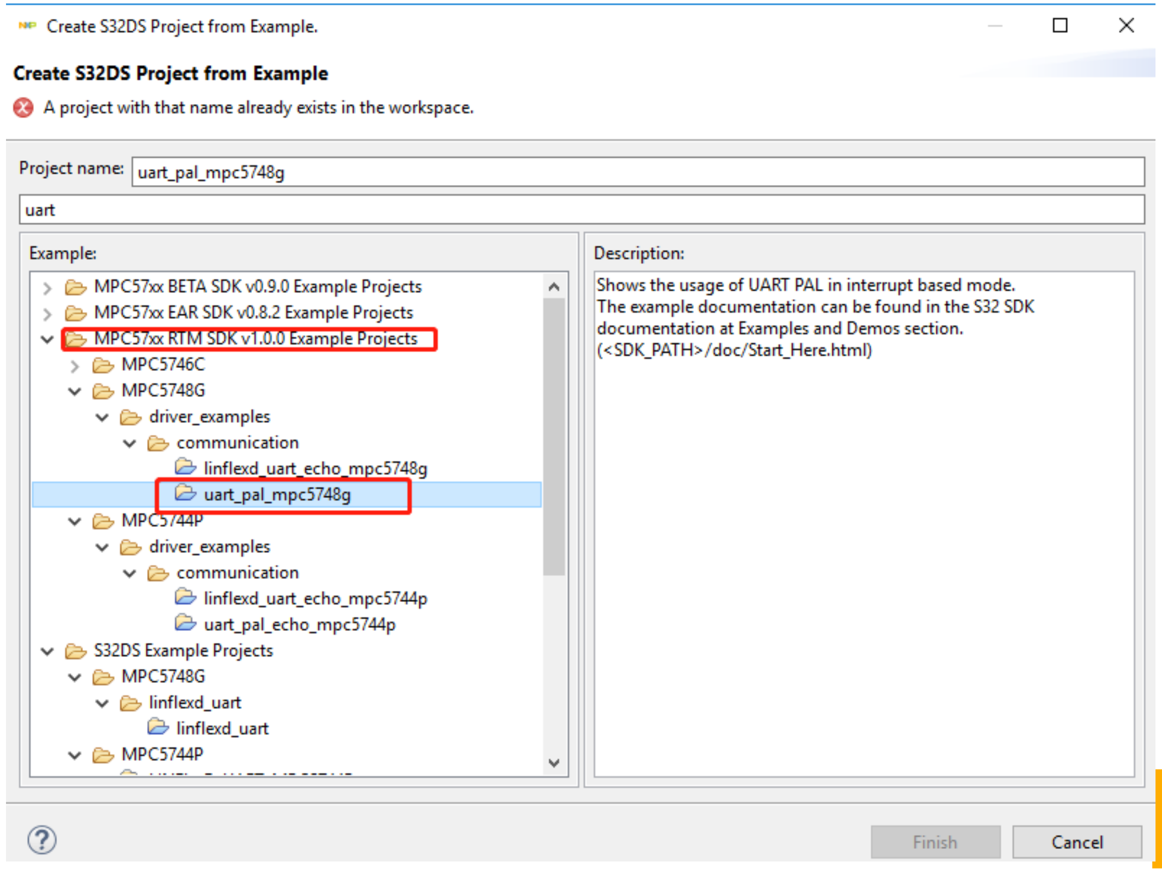
Hands-on – UART: Theory

- Full-duplex communication
- Separate clock for baud rate calculation
- The relationship “(2/3)* LIN_CLK > PBRIDGEx_CLK > 1/3*LIN_CLK” should be maintained.
- 15/16/7/8 bits data, parity
- 1/2/3 stop bits
- 12-bit + parity reception
- 4-byte buffer for reception, 4-byte buffer for transmission
- 12-bit counter for timeout management
- The maximum baud rate achievable is LIN_CLK/4 Mbit/s.
- For bit rate ≤ LIN_CLK/16 Mbit/s
- Sixteen times oversampling
- 3:1 majority voting
- For LIN_CLK/16 Mbit/s < bit rate ≤ LIN_CLK/8 Mbit/s
- Reduced over sampling programmable by the user
- 3:1 majority voting for reduced over sampling of 8
- For LIN_CLK/8 Mbit/s < bit rate ≤ LIN_CLK/4 Mbit/s
- Reduced over sampling programmable by the user

Hands-on – UART: Import Example Project

- Import 'uart' example provided with the SDK:
  - File->New->New S32DS Project from Example
  - Select: **uart_pal_mpc5748g** from **MPC57xxRTM SDK v1.0.0 Example Projects**




Create S32DS Project from Example.

Create S32DS Project from Example

A project with that name already exists in the workspace.

Project name: uart_pal_mpc5748g

uart

Example:

- MPC57xx BETA SDK v0.9.0 Example Projects
- MPC57xx EAR SDK v0.8.2 Example Projects
- MPC57xx RTM SDK v1.0.0 Example Projects
  - MPC5746C
  - MPC5748G
    - driver_examples
      - communication
        - linflexd_uart_echo_mpc5748g
        - uart_pal_mpc5748g
  - MPC5744P
    - driver_examples
      - communication
        - linflexd_uart_echo_mpc5744p
        - uart_pal_echo_mpc5744p
- S32DS Example Projects
  - MPC5748G
    - linflexd_uart
      - linflexd_uart
  - MPC5744P

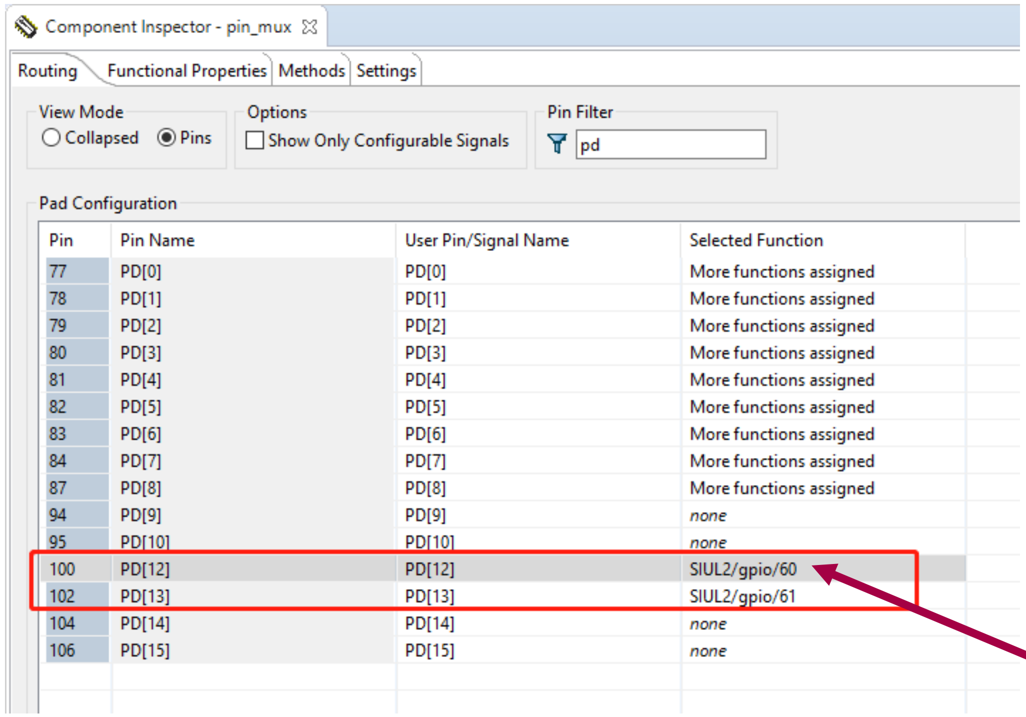
Description:

Shows the usage of UART PAL in interrupt based mode.
The example documentation can be found in the S32 SDK documentation at Examples and Demos section.
(<SDK_PATH>/doc/Start_Here.html)

Finish Cancel

Hands-on – UART: Modify-Peripheral Power Supply

- Enable the peripheral power supply
- Open ‘pin_mux’ component in ‘Component Inspector’ to configure pin routing
- SIUL2 tab -> GPIO 60 （61） > select the pin (one option) + direction output



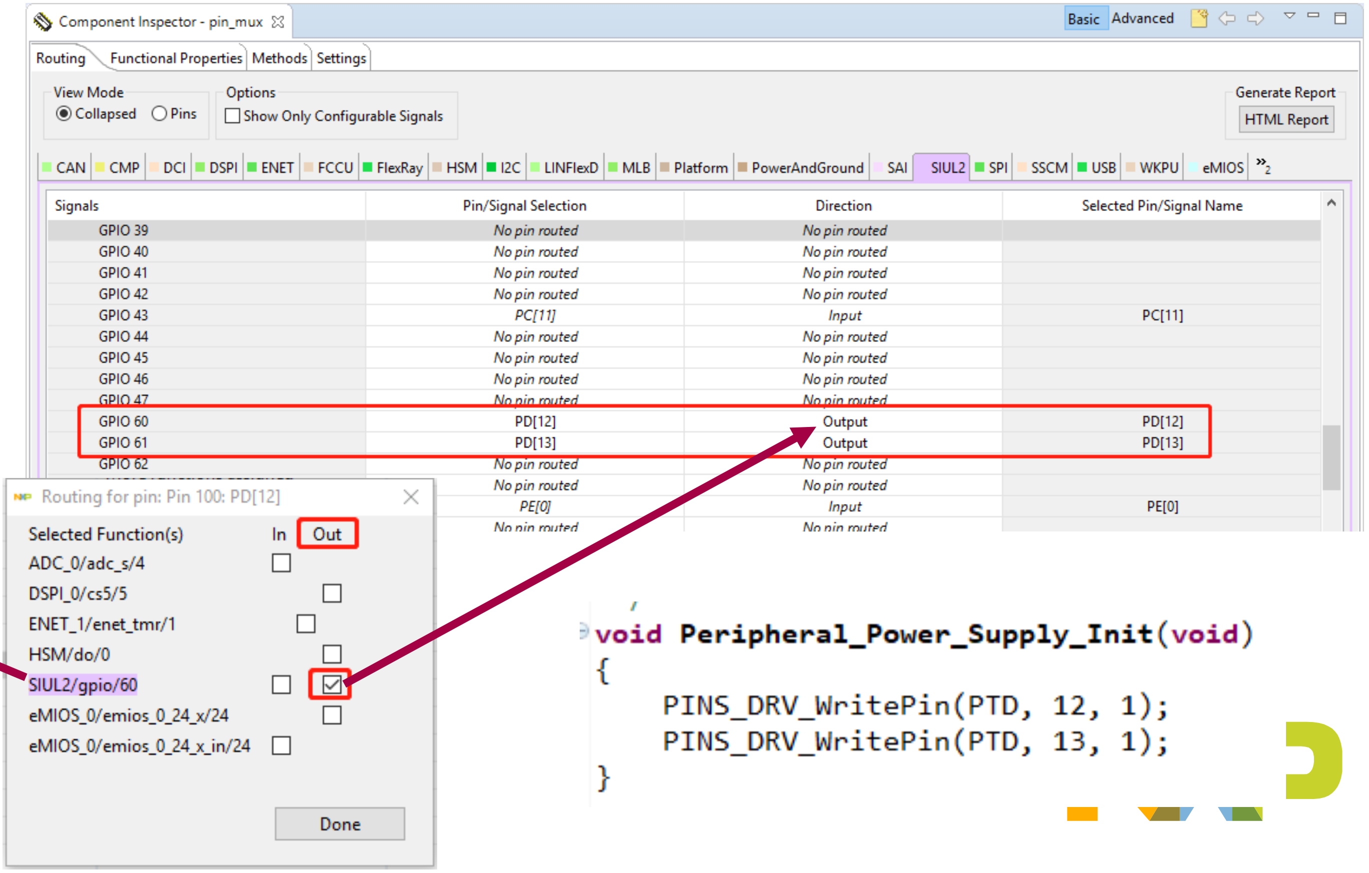
Component Inspector - pin_mux

Routing | Functional Properties | Methods | Settings

View Mode: ○ Collapsed [✓ ● Pins]
Options: ☐ Show Only Configurable Signals
Pin Filter: pd

Pad Configuration

Pin	Pin Name	User Pin/Signal Name	Selected Function
77	PD[0]	PD[0]	More functions assigned
78	PD[1]	PD[1]	More functions assigned
79	PD[2]	PD[2]	More functions assigned
80	PD[3]	PD[3]	More functions assigned
81	PD[4]	PD[4]	More functions assigned
82	PD[5]	PD[5]	More functions assigned
83	PD[6]	PD[6]	More functions assigned
84	PD[7]	PD[7]	More functions assigned
87	PD[8]	PD[8]	More functions assigned
94	PD[9]	PD[9]	*none*
95	PD[10]	PD[10]	*none*
100	PD[12]	PD[12]	SIUL2/gpio/60
102	PD[13]	PD[13]	SIUL2/gpio/61
104	PD[14]	PD[14]	*none*
106	PD[15]	PD[15]	*none*



Component Inspector - pin_mux

Basic | Advanced

Routing | Functional Properties | Methods | Settings

View Mode: [✓ ● Collapsed] ○ Pins
Options: ☐ Show Only Configurable Signals
Generate Report: HTML Report

CAN | CMP | DCI | DSPI | ENET | FCCU | FlexRay | HSM | I2C | LINFlexD | MLB | Platform | PowerAndGround | SAI | SIUL2 | SPI | SSCM | USB | WKPU | eMIOS

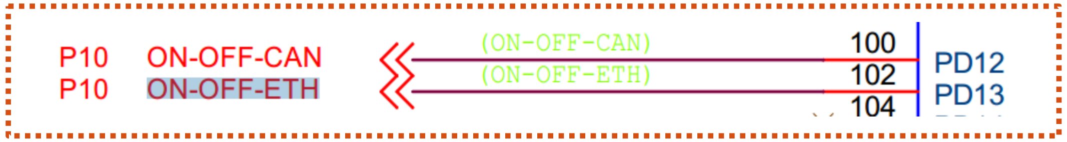
Signals	Pin/Signal Selection	Direction	Selected Pin/Signal Name
GPIO 39	*No pin routed*	*No pin routed*	
GPIO 40	*No pin routed*	*No pin routed*	
GPIO 41	*No pin routed*	*No pin routed*	
GPIO 42	*No pin routed*	*No pin routed*	
GPIO 43	*PC[11]*	*Input*	PC[11]
GPIO 44	*No pin routed*	*No pin routed*	
GPIO 45	*No pin routed*	*No pin routed*	
GPIO 46	*No pin routed*	*No pin routed*	
GPIO 47	*No pin routed*	*No pin routed*	
GPIO 60	PD[12]	Output	PD[12]
GPIO 61	PD[13]	Output	PD[13]
GPIO 62	*No pin routed*	*No pin routed*	
	No pin routed	*No pin routed*	
	PE[0]	*Input*	PE[0]
	No pin routed	*No pin routed*	

Routing for pin: Pin 100: PD[12]

Selected Function(s)	In	Out
ADC_0/adc_s/4	☐	
DSPI_0/cs5/5		☐
ENET_1/enet_tmr/1	☐	
HSM/do/0		☐
SIUL2/gpio/60	☐	☑
eMIOS_0/emios_0_24_x/24		☐
eMIOS_0/emios_0_24_x_in/24	☐	

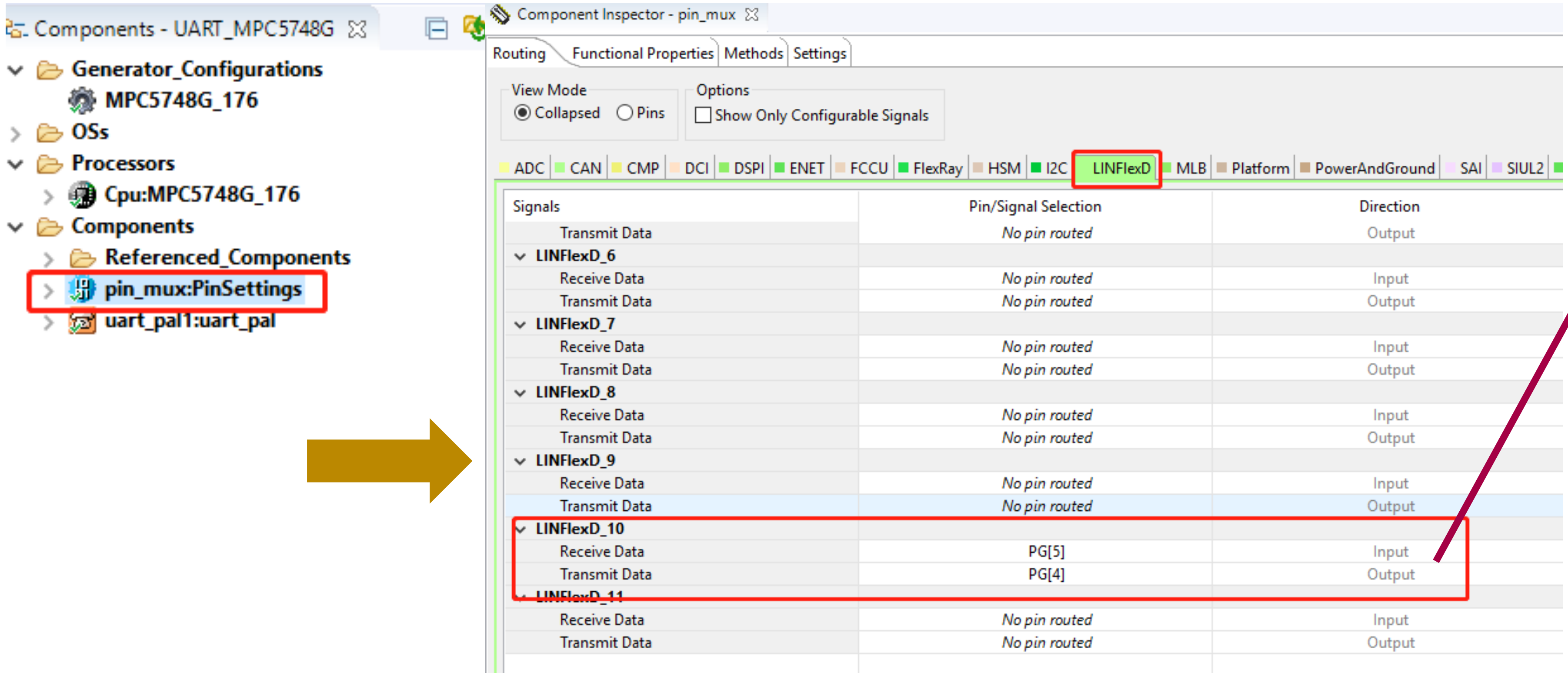
Done

```
void Peripheral_Power_Supply_Init(void)
{
    PINS_DRV_WritePin(PTD, 12, 1);
    PINS_DRV_WritePin(PTD, 13, 1);
}
```



Hands-on – UART: UART configuration

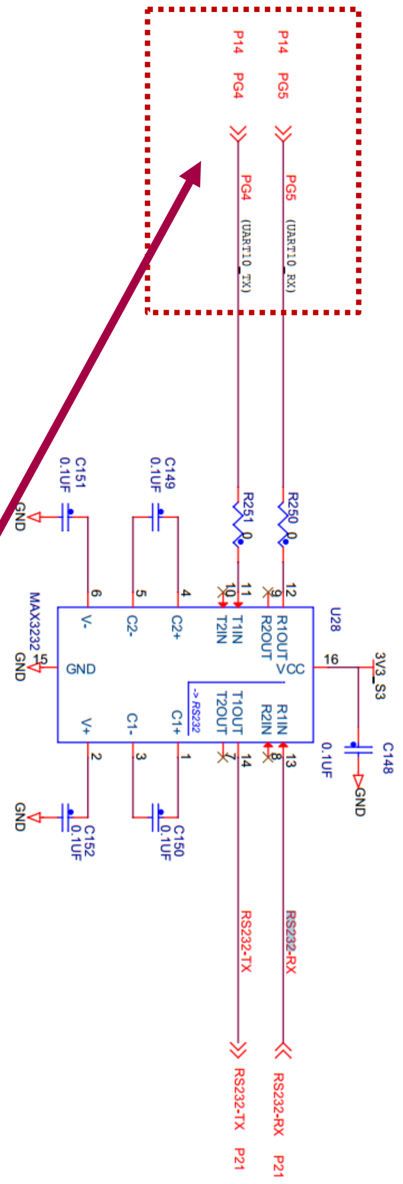
(1) Configure the port of UART



View Mode: (● Collapsed) ○ Pins

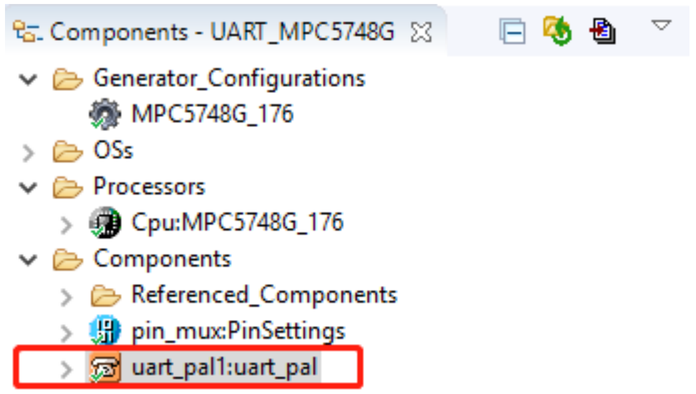
Options: ☐ Show Only Configurable Signals

Signals	Pin/Signal Selection	Direction
Transmit Data	*No pin routed*	Output
LINFlexD_6		
Receive Data	*No pin routed*	Input
Transmit Data	*No pin routed*	Output
LINFlexD_7		
Receive Data	*No pin routed*	Input
Transmit Data	*No pin routed*	Output
LINFlexD_8		
Receive Data	*No pin routed*	Input
Transmit Data	*No pin routed*	Output
LINFlexD_9		
Receive Data	*No pin routed*	Input
Transmit Data	*No pin routed*	Output
LINFlexD_10		
Receive Data	PG[5]	Input
Transmit Data	PG[4]	Output
LINFlexD_11		
Receive Data	*No pin routed*	Input
Transmit Data	*No pin routed*	Output

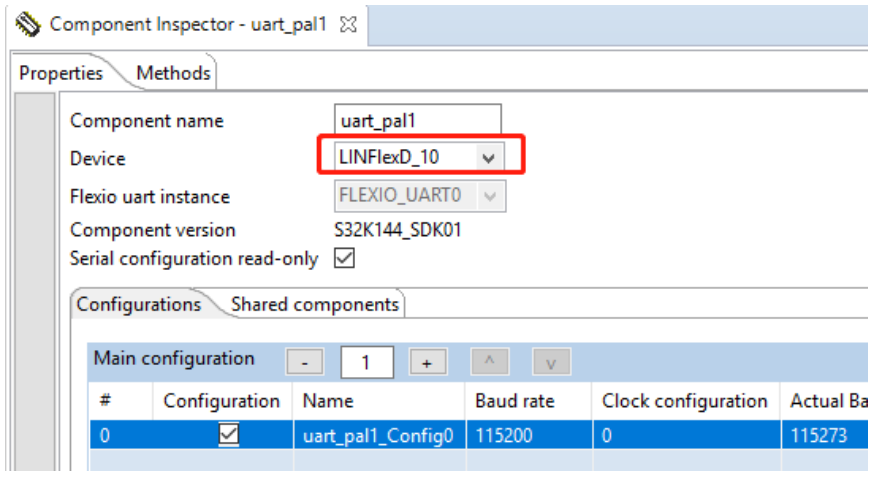


Hands-on – UART: UART configuration

(2) Configuration of UART properties


Components - UART_MPC5748G

- Generator_Configurations
  - MPC5748G_176
- OSs
- Processors
  - Cpu:MPC5748G_176
- Components
  - Referenced_Components
  - pin_mux:PinSettings
  - uart_pal1:uart_pal


Component Inspector - uart_pal1

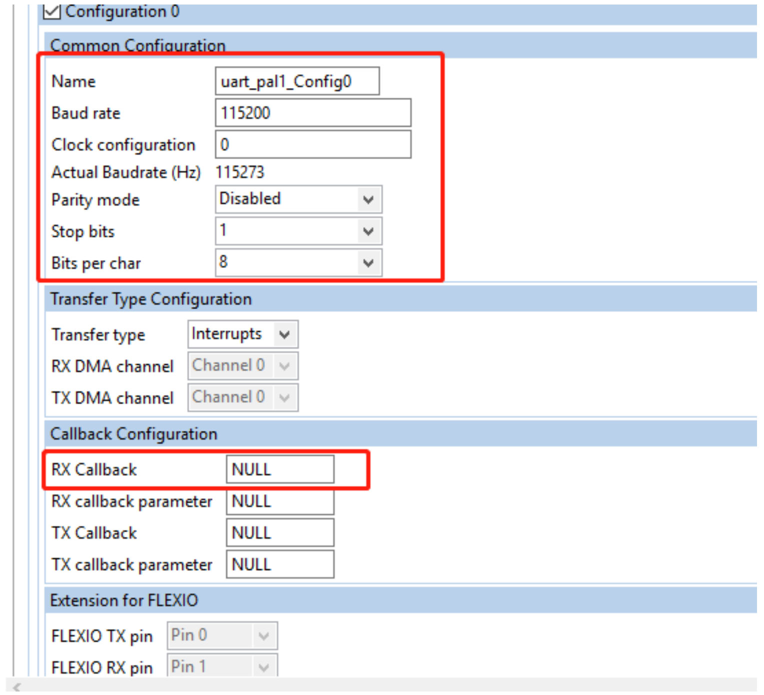
Properties | Methods

Component name	uart_pal1
Device	LINFlexD_10
Flexio uart instance	FLEXIO_UART0
Component version	S32K144_SDK01
Serial configuration read-only	✓

Configurations | Shared components

Main configuration - 1 +

#	Configuration	Name	Baud rate	Clock configuration	Actual Ba
0	✓	uart_pal1_Config0	115200	0	115273


Configuration 0

Common Configuration

Name	uart_pal1_Config0
Baud rate	115200
Clock configuration	0
Actual Baudrate (Hz)	115273
Parity mode	Disabled
Stop bits	1
Bits per char	8

Transfer Type Configuration

Transfer type	Interrupts
RX DMA channel	Channel 0
TX DMA channel	Channel 0

Callback Configuration

RX Callback	NULL
RX callback parameter	NULL
TX Callback	NULL
TX callback parameter	NULL

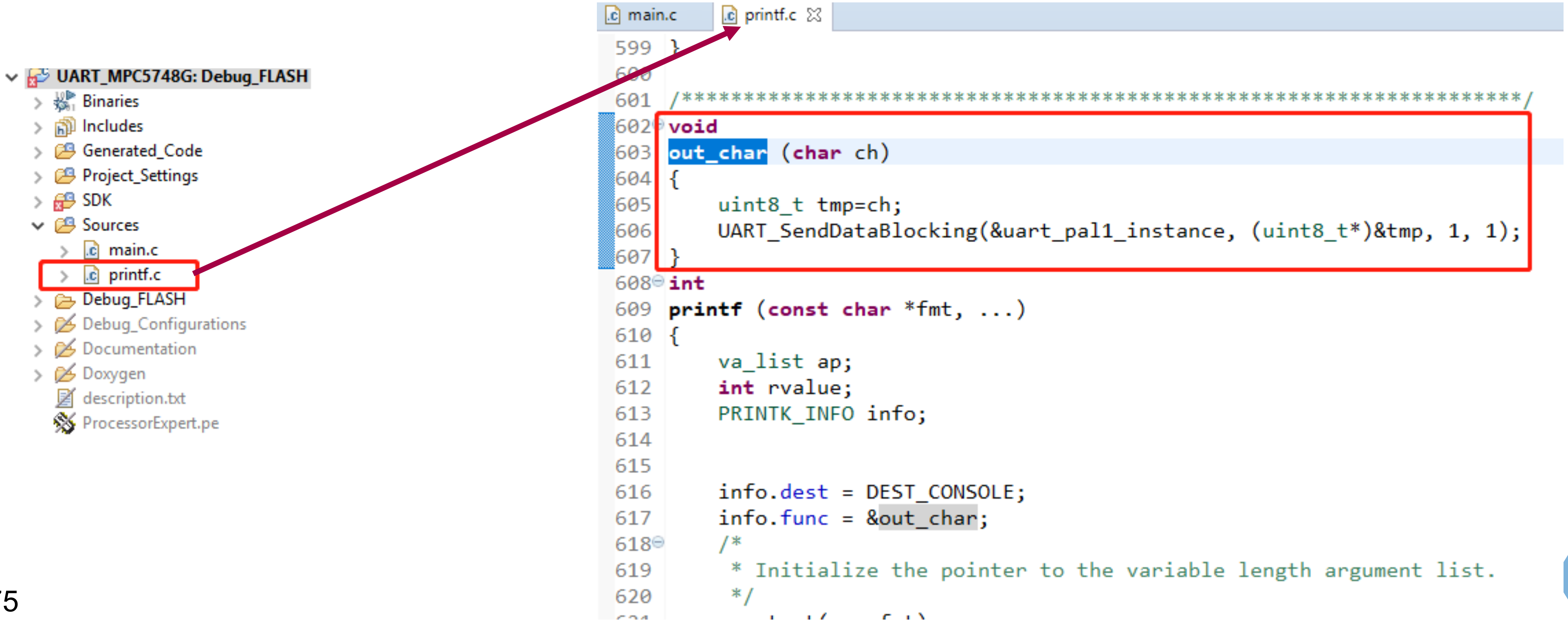
Extension for FLEXIO

FLEXIO TX pin	Pin 0
FLEXIO RX pin	Pin 1

Hands-on – UART: UART configuration

(3) Port printf library functions

- You can refer to the sample demo(The project of *UART_MPC5748G*) we provided. And copy the *printf.c* file to your project without any modification.



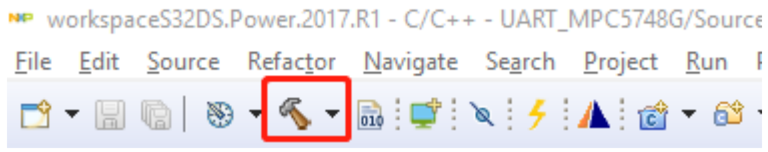
UART_MPC5748G: Debug_FLASH
- Binaries
- Includes
- Generated_Code
- Project_Settings
- SDK
- Sources
  - main.c
  - printf.c
- Debug_FLASH
- Debug_Configurations
- Documentation
- Doxygen
- description.txt
- ProcessorExpert.pe

main.c | printf.c

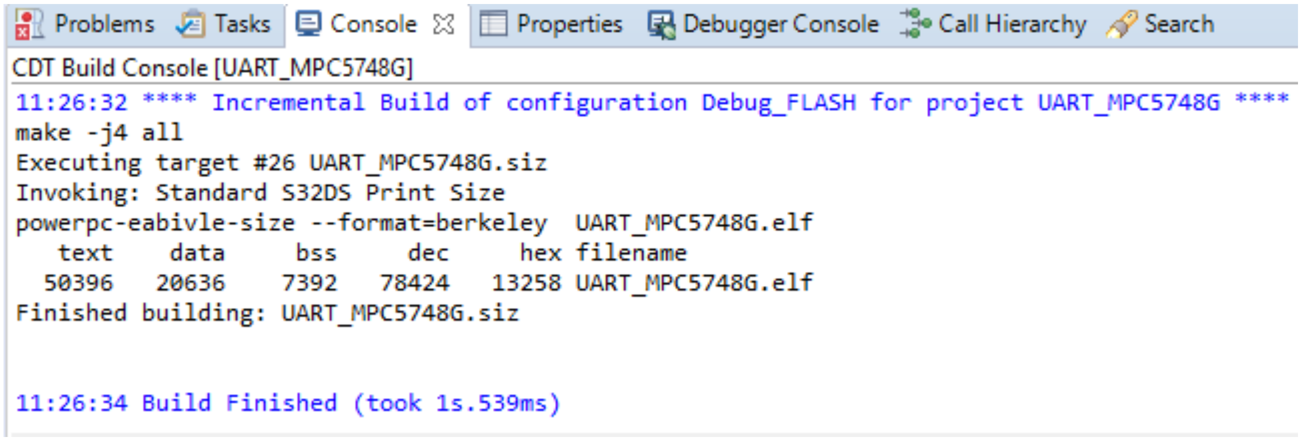
```
}

/*********************************************************************/
void
out_char (char ch)
{
    uint8_t tmp=ch;
    UART_SendDataBlocking(&uart_pal1_instance, (uint8_t*)&tmp, 1, 1);
}
int
printf (const char *fmt, ...)
{
    va_list ap;
    int rvalue;
    PRINTK_INFO info;


    info.dest = DEST_CONSOLE;
    info.func = &out_char;
    /*
     * Initialize the pointer to the variable length argument list.
     */
```

Hands-on – UART: Build and Debug

- Click the ‘build project’ button – make sure there are no compilation errors



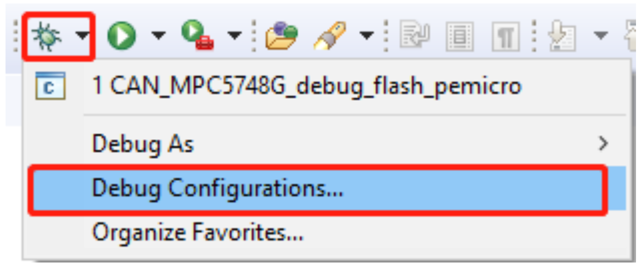
workspaceS32DS.Power.2017.R1 - C/C++ - UART_MPC5748G/Source
File Edit Source Refactor Navigate Search Project Run

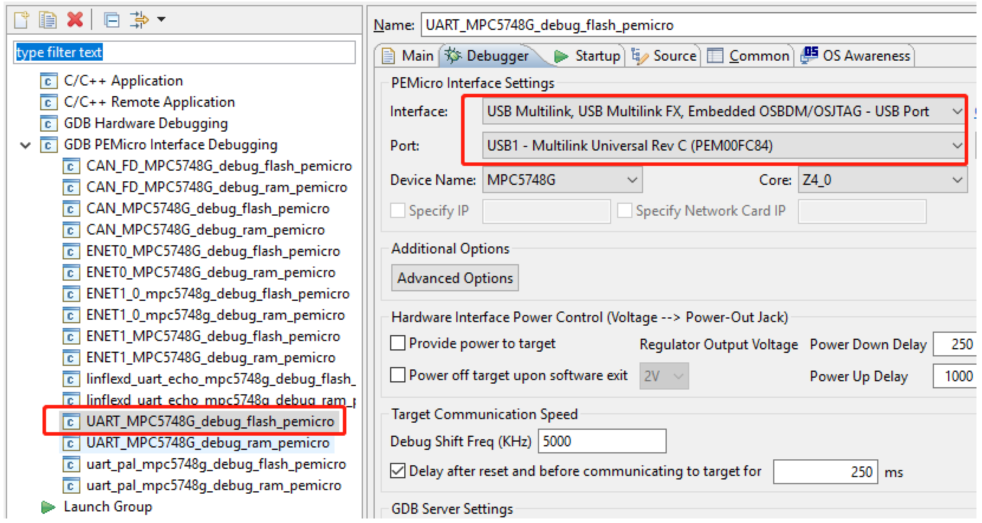


Problems Tasks Console Properties Debugger Console Call Hierarchy Search

```
CDT Build Console [UART_MPC5748G]
11:26:32 **** Incremental Build of configuration Debug_FLASH for project UART_MPC5748G ****
make -j4 all
Executing target #26 UART_MPC5748G.siz
Invoking: Standard S32DS Print Size
powerpc-eabivle-size --format=berkeley  UART_MPC5748G.elf
   text    data     bss     dec     hex filename
  50396   20636    7392   78424   13258 UART_MPC5748G.elf
Finished building: UART_MPC5748G.siz


11:26:34 Build Finished (took 1s.539ms)
```

- Select the correct debug configuration and interface to debug the application





type filter text
C/C++ Application
C/C++ Remote Application
GDB Hardware Debugging
GDB PEMicro Interface Debugging
CAN_FD_MPC5748G_debug_flash_pemicro
CAN_FD_MPC5748G_debug_ram_pemicro
CAN_MPC5748G_debug_flash_pemicro
CAN_MPC5748G_debug_ram_pemicro
ENET0_MPC5748G_debug_flash_pemicro
ENET0_MPC5748G_debug_ram_pemicro
ENET1_0_mpc5748g_debug_flash_pemicro
ENET1_0_mpc5748g_debug_ram_pemicro
ENET1_MPC5748G_debug_flash_pemicro
ENET1_MPC5748G_debug_ram_pemicro
linflexd_uart_echo_mpc5748g_debug_flash_
linflexd_uart_echo_mpc5748g_debug_ram_
UART_MPC5748G_debug_flash_pemicro
UART_MPC5748G_debug_ram_pemicro
uart_pal_mpc5748g_debug_flash_pemicro
uart_pal_mpc5748g_debug_ram_pemicro
Launch Group
Main Debugger Startup Source Common OS Awareness
Advanced Options

Name: UART_MPC5748G_debug_flash_pemicro

PEMicro Interface Settings

Interface: USB Multilink, USB Multilink FX, Embedded OSBDM/OSJTAG - USB Port

Port: USB1 - Multilink Universal Rev C (PEM00FC84)

Device Name: MPC5748G Core: Z4_0

☐ Specify IP ☐ Specify Network Card IP

Additional Options

Hardware Interface Power Control (Voltage --> Power-Out Jack)

☐ Provide power to target Regulator Output Voltage Power Down Delay 250

☐ Power off target upon software exit 2V Power Up Delay 1000

Target Communication Speed

Debug Shift Freq (KHz) 5000

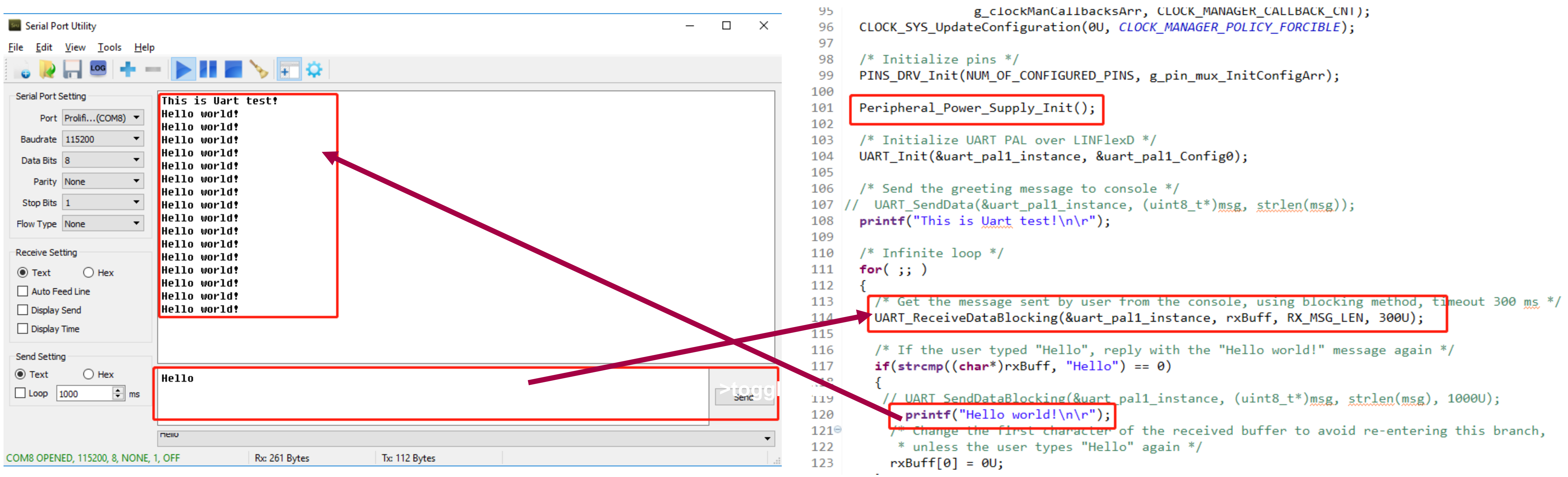
☑ Delay after reset and before communicating to target for 250 ms

GDB Server Settings

Hands-on – UART: Build and Debug

TEST: UART send and receive

Main.c



Serial Port Setting
- Port: Prolifi...(COM8)
- Baudrate: 115200
- Data Bits: 8
- Parity: None
- Stop Bits: 1
- Flow Type: None

Receive Setting
- [✓ Text] Hex
- Auto Feed Line
- Display Send
- Display Time

Send Setting
- [✓ Text] Hex
- Loop 1000 ms

```
This is Uart test!
Hello world!
Hello world!
Hello world!
Hello world!
Hello world!
Hello world!
Hello world!
Hello world!
Hello world!
Hello world!
Hello world!
Hello world!
Hello world!
Hello world!
Hello world!
Hello world!
```

```
Hello
```

COM8 OPENED, 115200, 8, NONE, 1, OFF   Rx: 261 Bytes   Tx: 112 Bytes

```c
            g_clockManCallbacksArr, CLOCK_MANAGER_CALLBACK_CNT);
CLOCK_SYS_UpdateConfiguration(0U, CLOCK_MANAGER_POLICY_FORCIBLE);

/* Initialize pins */
PINS_DRV_Init(NUM_OF_CONFIGURED_PINS, g_pin_mux_InitConfigArr);

Peripheral_Power_Supply_Init();

/* Initialize UART PAL over LINFlexD */
UART_Init(&uart_pal1_instance, &uart_pal1_Config0);

/* Send the greeting message to console */
//  UART_SendData(&uart_pal1_instance, (uint8_t*)msg, strlen(msg));
printf("This is Uart test!\n\r");

/* Infinite loop */
for( ;; )
{
  /* Get the message sent by user from the console, using blocking method, timeout 300 ms */
  UART_ReceiveDataBlocking(&uart_pal1_instance, rxBuff, RX_MSG_LEN, 300U);

  /* If the user typed "Hello", reply with the "Hello world!" message again */
  if(strcmp((char*)rxBuff, "Hello") == 0)
  {
    // UART_SendDataBlocking(&uart_pal1_instance, (uint8_t*)msg, strlen(msg), 1000U);
    printf("Hello world!\n\r");
    /* Change the first character of the received buffer to avoid re-entering this branch,
     * unless the user types "Hello" again */
    rxBuff[0] = 0U;
```

06.

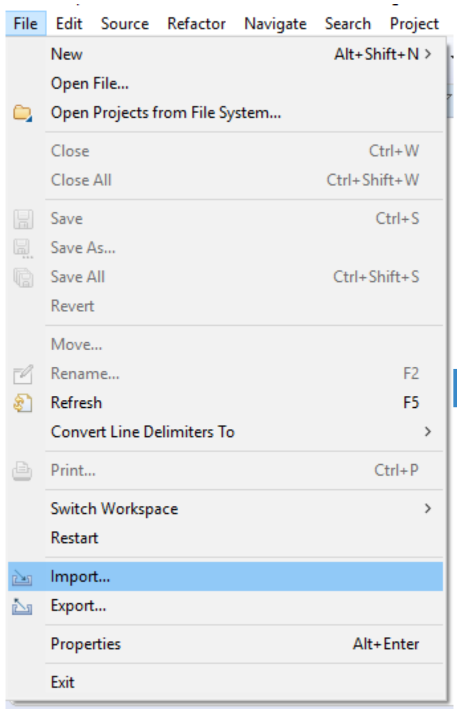
Hands-on – LIN

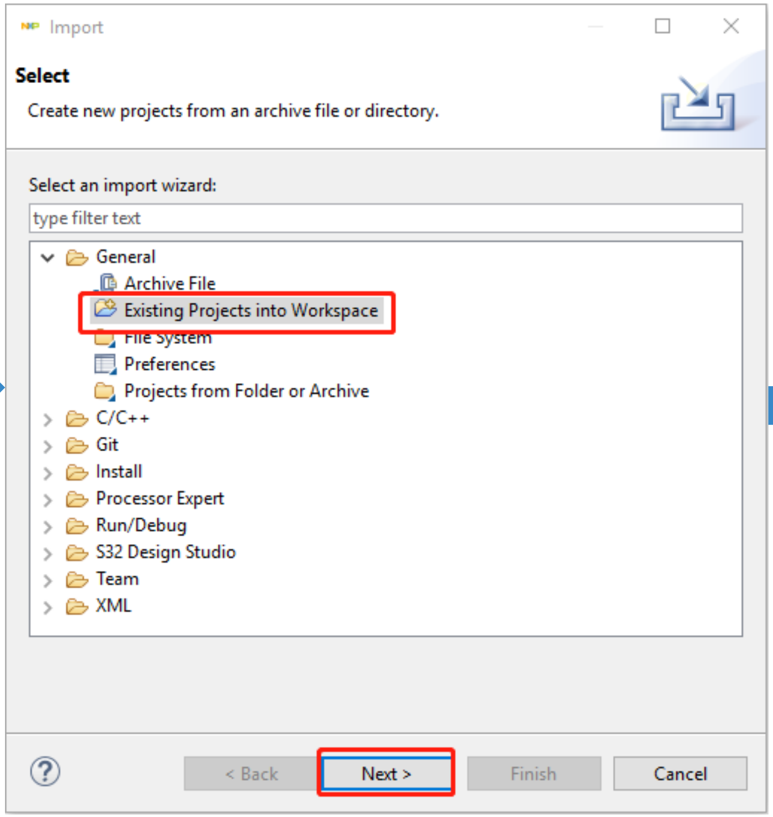
Hands-on – LIN: Theory

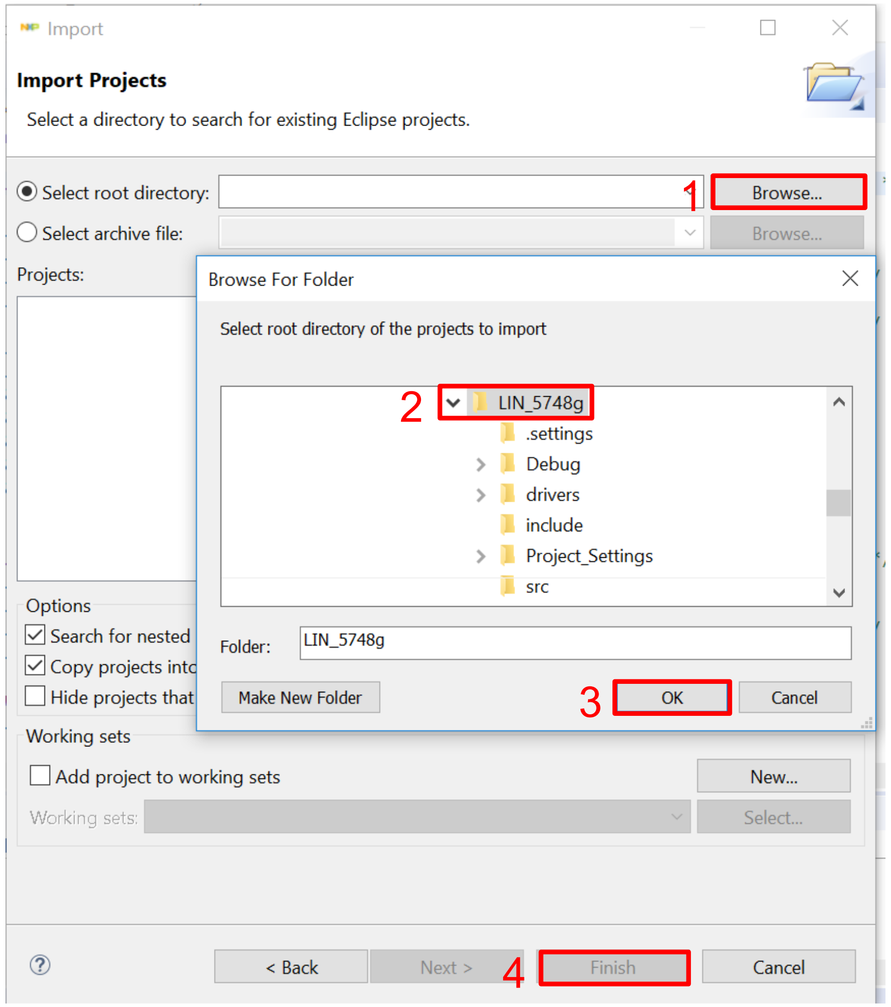
- Supports LIN protocol version 1.3, 2.0, 2.1, and 2.2
- Bit rates up to 20 Kbit/s (LIN protocol)
- Master/Slave mode
- Classic and Enhanced Checksum calculation and check
- Single 8-byte buffer or FIFO for Transmission/Reception
- Timeout management
- Identifier filters
- DMA interface
- Supports a maximum of 16 possible identifiers
- Master mode with autonomous message handling
- Wakeup event on dominant bit detection
- True LIN field state machine
- Advanced LIN error detection
- Header, response, and frame timeout
- Slave mode
- Autonomous header handling
- Autonomous transmit/receive data handling
- Identifier filters for autonomous message handling in Slave mode
- Separate clock for baud rate calculation
- The relationship “(2/3)* LIN_CLK > PBRIDGEx_CLK > 1/3*LIN_CLK”
- should be maintained.

Hands-on – LIN: Import Existing Project

- Import existing S32DS project for MPC5748G:



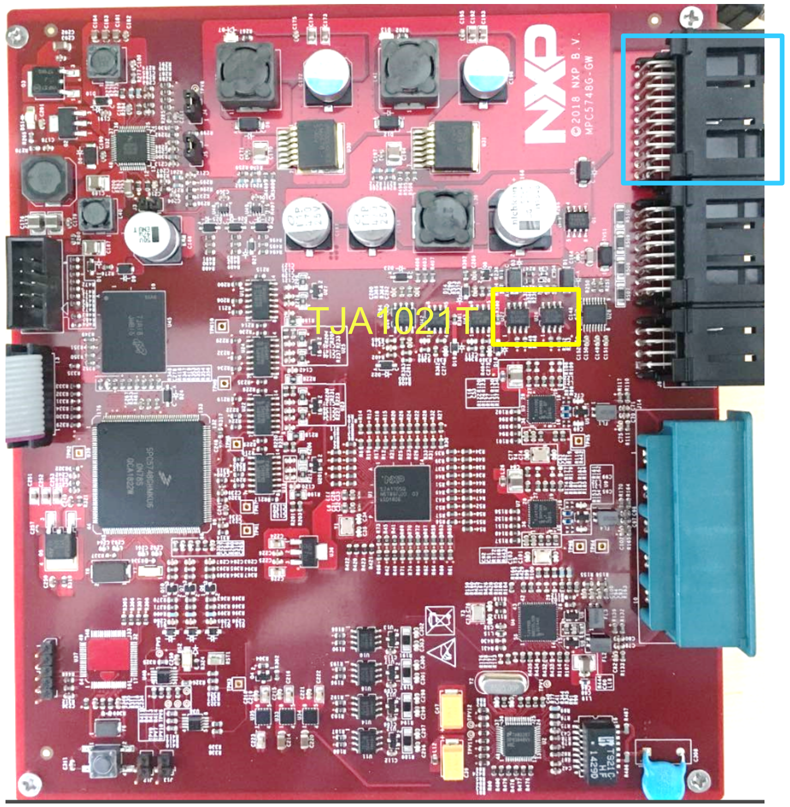


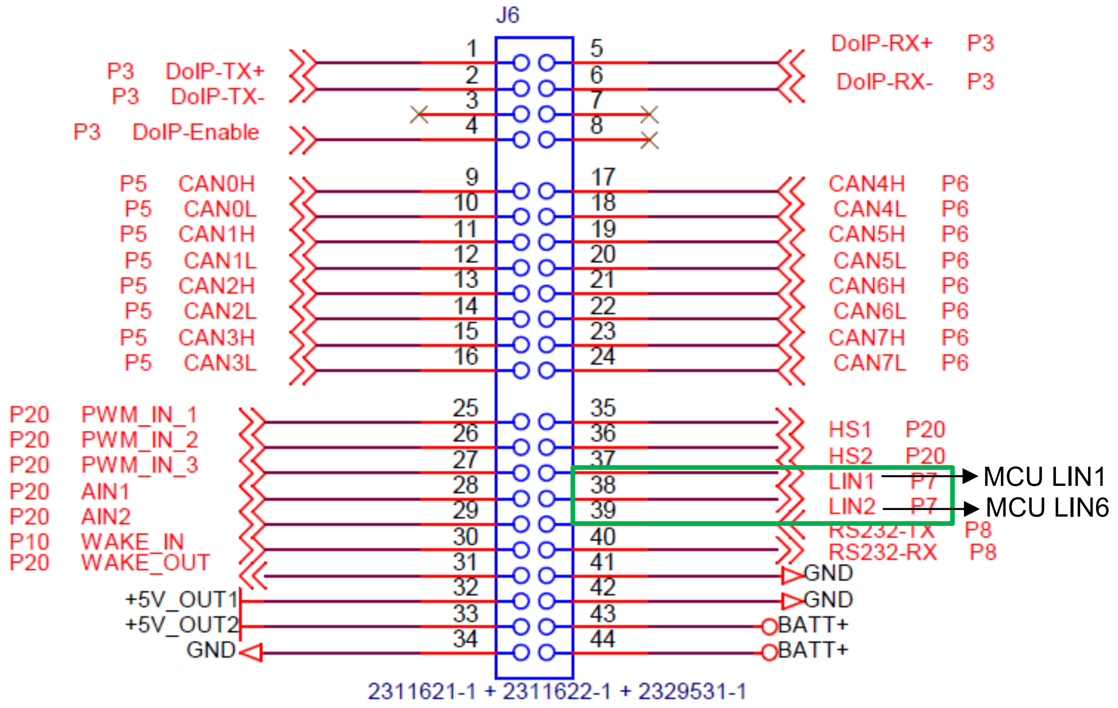


Then the selected project will be added to the Project Explorer window

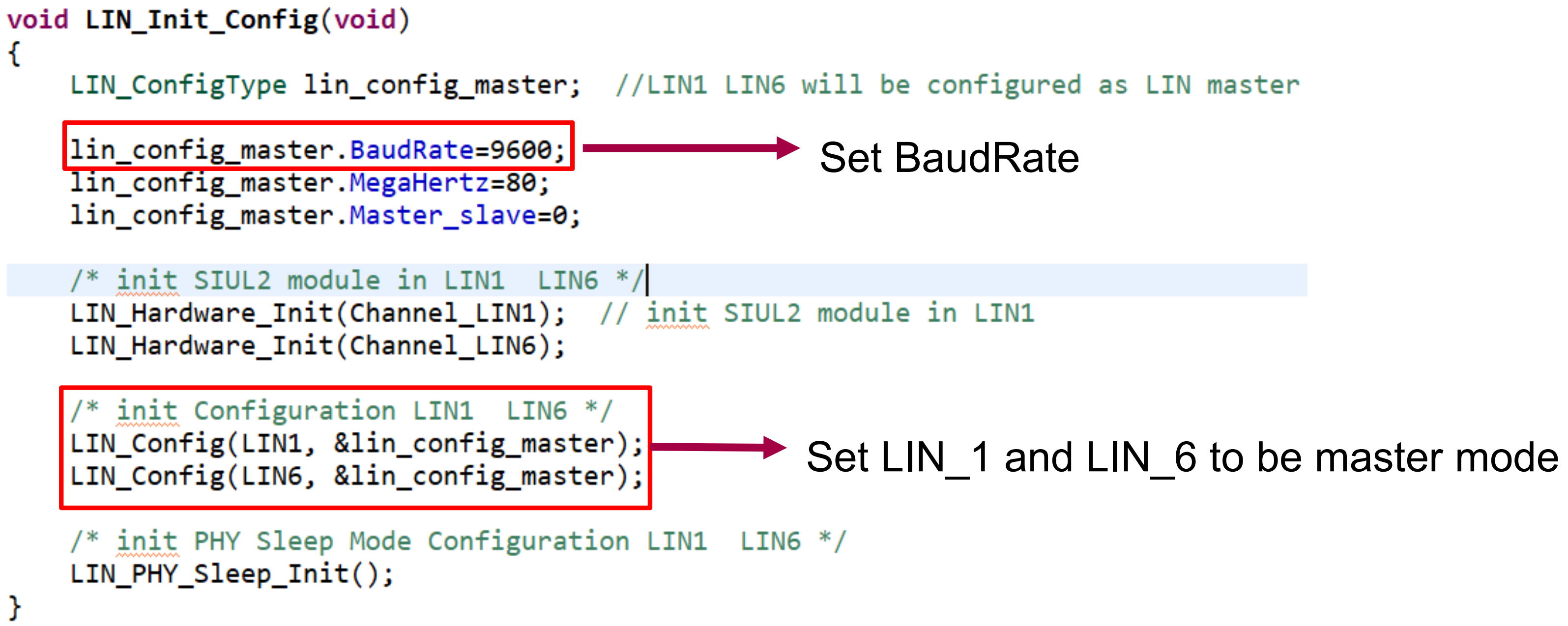
Hands-on – LIN: Resources

- Resources to be used:
  - on-board user LIN ports (hardwired to GPIOs)



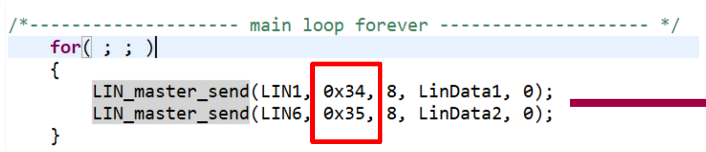


Hands-on – LIN: Application Code



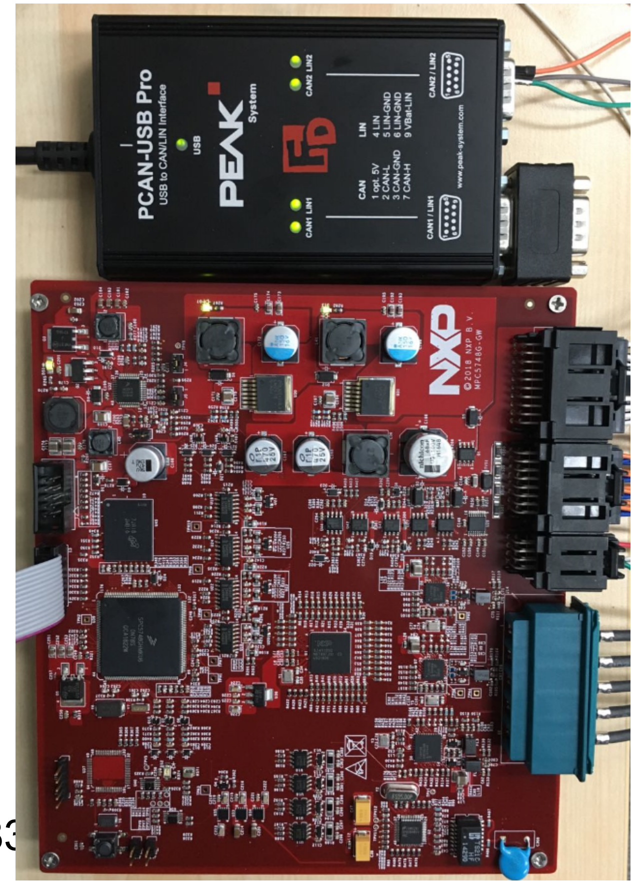
```
void LIN_Init_Config(void)
{
    LIN_ConfigType lin_config_master;  //LIN1 LIN6 will be configured as LIN master

    lin_config_master.BaudRate=9600;
    lin_config_master.MegaHertz=80;
    lin_config_master.Master_slave=0;

    /* init SIUL2 module in LIN1  LIN6 */
    LIN_Hardware_Init(Channel_LIN1);  // init SIUL2 module in LIN1
    LIN_Hardware_Init(Channel_LIN6);

    /* init Configuration LIN1  LIN6 */
    LIN_Config(LIN1, &lin_config_master);
    LIN_Config(LIN6, &lin_config_master);

    /* init PHY Sleep Mode Configuration LIN1  LIN6 */
    LIN_PHY_Sleep_Init();
}
```

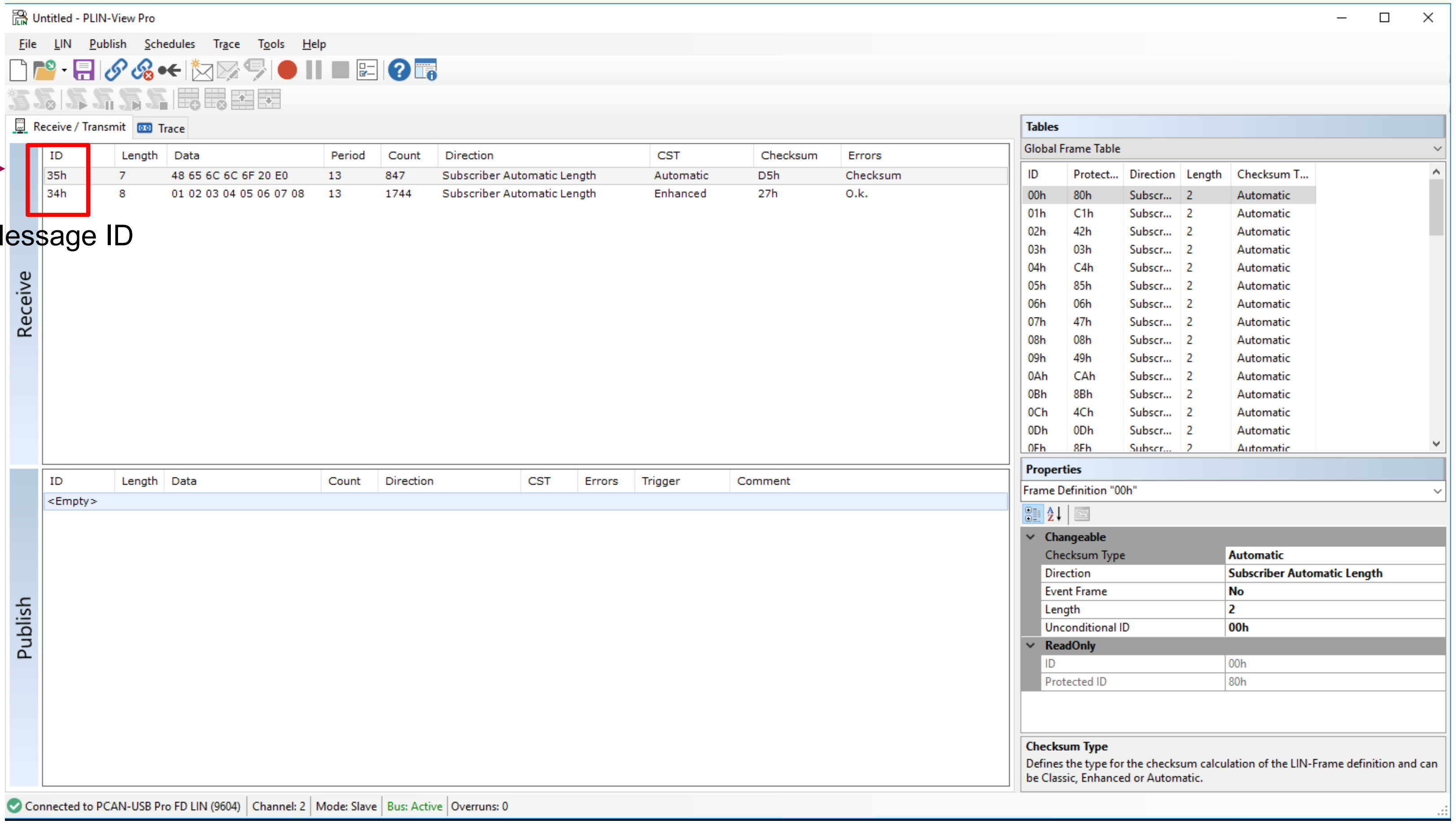
Hands-on – LIN: Build and Debug



```
/*-------------------- main loop forever -------------------- */
    for( ; ; )
    {
        LIN_master_send(LIN1, 0x34, 8, LinData1, 0);
        LIN_master_send(LIN6, 0x35, 8, LinData2, 0);
    }
```

Message ID





Message ID

Untitled - PLIN-View Pro

ID	Length	Data	Period	Count	Direction	CST	Checksum	Errors
35h	7	48 65 6C 6C 6F 20 E0	13	847	Subscriber Automatic Length	Automatic	D5h	Checksum
34h	8	01 02 03 04 05 06 07 08	13	1744	Subscriber Automatic Length	Enhanced	27h	O.k.

ID	Length	Data	Count	Direction	CST	Errors	Trigger	Comment
<Empty>								

Tables

Global Frame Table

ID	Protect...	Direction	Length	Checksum T...
00h	80h	Subscr...	2	Automatic
01h	C1h	Subscr...	2	Automatic
02h	42h	Subscr...	2	Automatic
03h	03h	Subscr...	2	Automatic
04h	C4h	Subscr...	2	Automatic
05h	85h	Subscr...	2	Automatic
06h	06h	Subscr...	2	Automatic
07h	47h	Subscr...	2	Automatic
08h	08h	Subscr...	2	Automatic
09h	49h	Subscr...	2	Automatic
0Ah	CAh	Subscr...	2	Automatic
0Bh	8Bh	Subscr...	2	Automatic
0Ch	4Ch	Subscr...	2	Automatic
0Dh	0Dh	Subscr...	2	Automatic
0Eh	8Eh	Subscr...	2	Automatic

Properties

Frame Definition "00h"

Changeable	
Checksum Type	**Automatic**
Direction	**Subscriber Automatic Length**
Event Frame	**No**
Length	**2**
Unconditional ID	**00h**
ReadOnly	
ID	00h
Protected ID	80h

Checksum Type
Defines the type for the checksum calculation of the LIN-Frame definition and can be Classic, Enhanced or Automatic.

Connected to PCAN-USB Pro FD LIN (9604) | Channel: 2 | Mode: Slave | Bus: Active | Overruns: 0

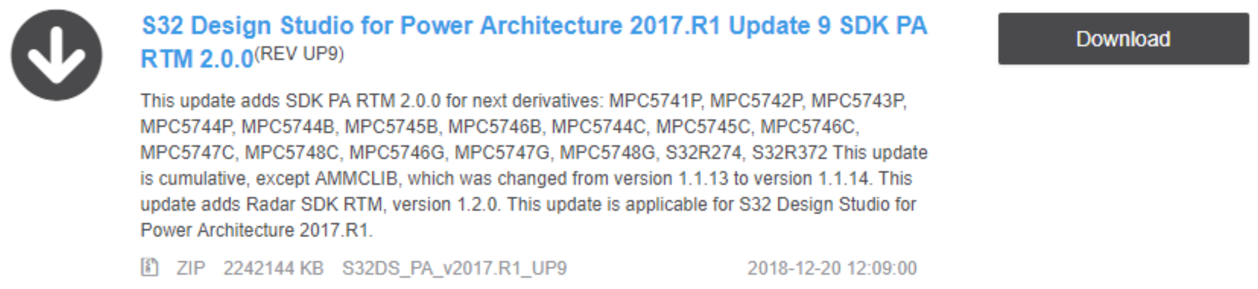
07.

Hands-on – eMMC+Fatfs

Hands-on – eMMC+Fatfs: Objective

- Features of UART module on MPC5748G
- How to set a pin as output/input with SDK
- How to configure the port of UART
- How to configure the port of eMMC
- How to modify an existing SDK project with S32DS to suit this board

NOTE: This demo is only available for the SDK version RTM2.0.0, Please make sure your SDK has been upgraded to RTM2.0.0

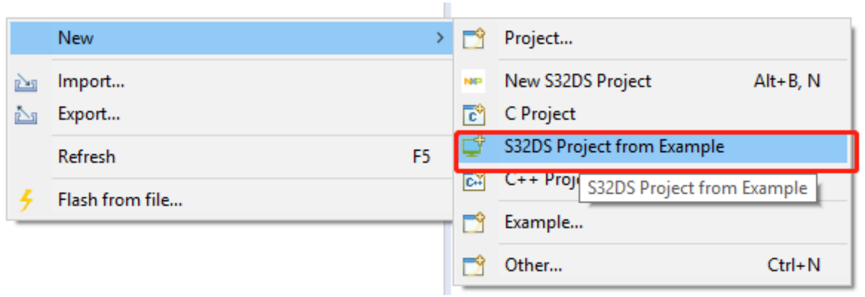

S32 Design Studio for Power Architecture 2017.R1 Update 9 SDK PA RTM 2.0.0 (REV UP9)

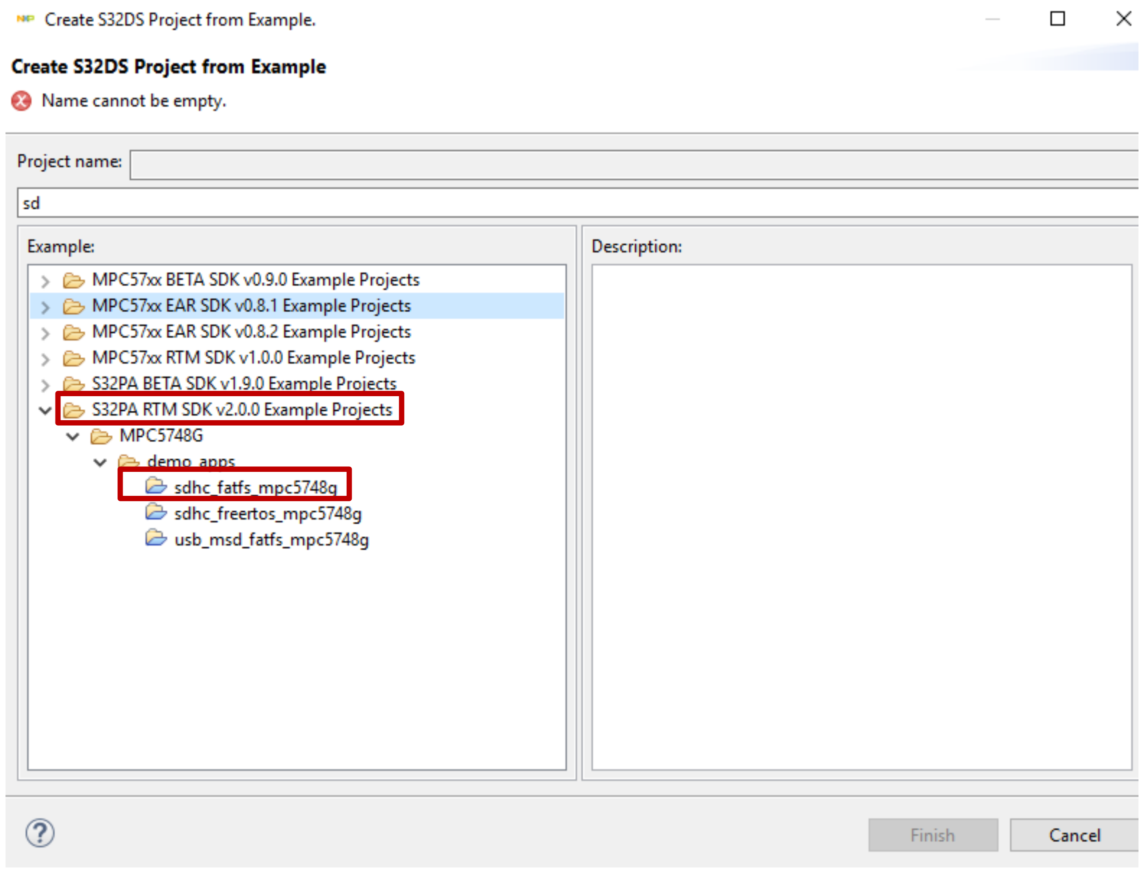
This update adds SDK PA RTM 2.0.0 for next derivatives: MPC5741P, MPC5742P, MPC5743P, MPC5744P, MPC5744B, MPC5745B, MPC5746B, MPC5744C, MPC5745C, MPC5746C, MPC5747C, MPC5748C, MPC5746G, MPC5747G, MPC5748G, S32R274, S32R372 This update is cumulative, except AMMCLIB, which was changed from version 1.1.13 to version 1.1.14. This update adds Radar SDK RTM, version 1.2.0. This update is applicable for S32 Design Studio for Power Architecture 2017.R1.

ZIP 2242144 KB S32DS_PA_v2017.R1_UP9 2018-12-20 12:09:00

Hands-on – eMMC+Fatfs: Import Example Project

- Import ‘sdhc_fatfs_mpc5748g’ example provided with the SDK:
  - File->New->New S32DS Project from Example
  - Select: **sdhc_fatfs _mpc5748g** from **MPC57xxRTM SDK v2.0.0 Example Projects**




Create S32DS Project from Example.

Create S32DS Project from Example

Name cannot be empty.

Project name:

sd

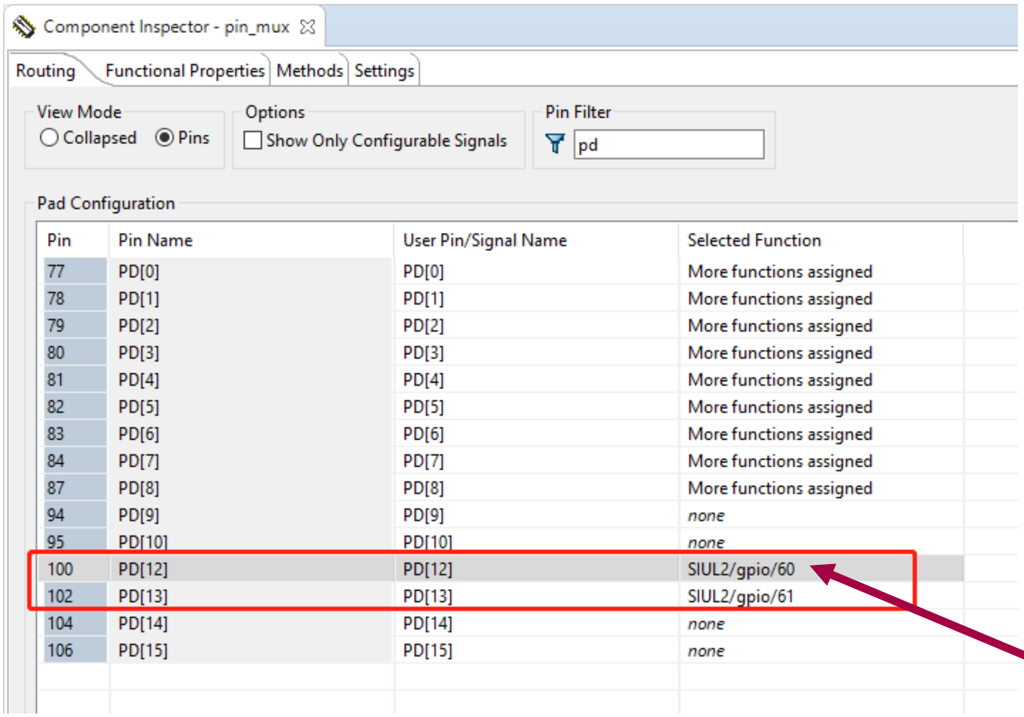
Example:

- MPC57xx BETA SDK v0.9.0 Example Projects
- MPC57xx EAR SDK v0.8.1 Example Projects
- MPC57xx EAR SDK v0.8.2 Example Projects
- MPC57xx RTM SDK v1.0.0 Example Projects
- S32PA BETA SDK v1.9.0 Example Projects
- S32PA RTM SDK v2.0.0 Example Projects
  - MPC5748G
    - demo_apps
      - sdhc_fatfs_mpc5748g
      - sdhc_freertos_mpc5748g
      - usb_msd_fatfs_mpc5748g

Description:

Hands-on – eMMC+Fatfs: Modify-Peripheral Power Supply

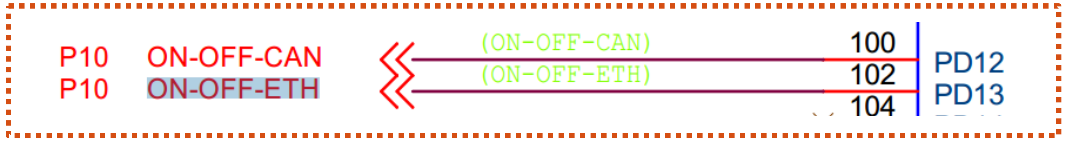
- Enable the peripheral power supply
- Open 'pin_mux' component in 'Component Inspector' to configure pin routing
- SIUL2 tab -> GPIO 60 （61） > select the pin (one option) + direction output

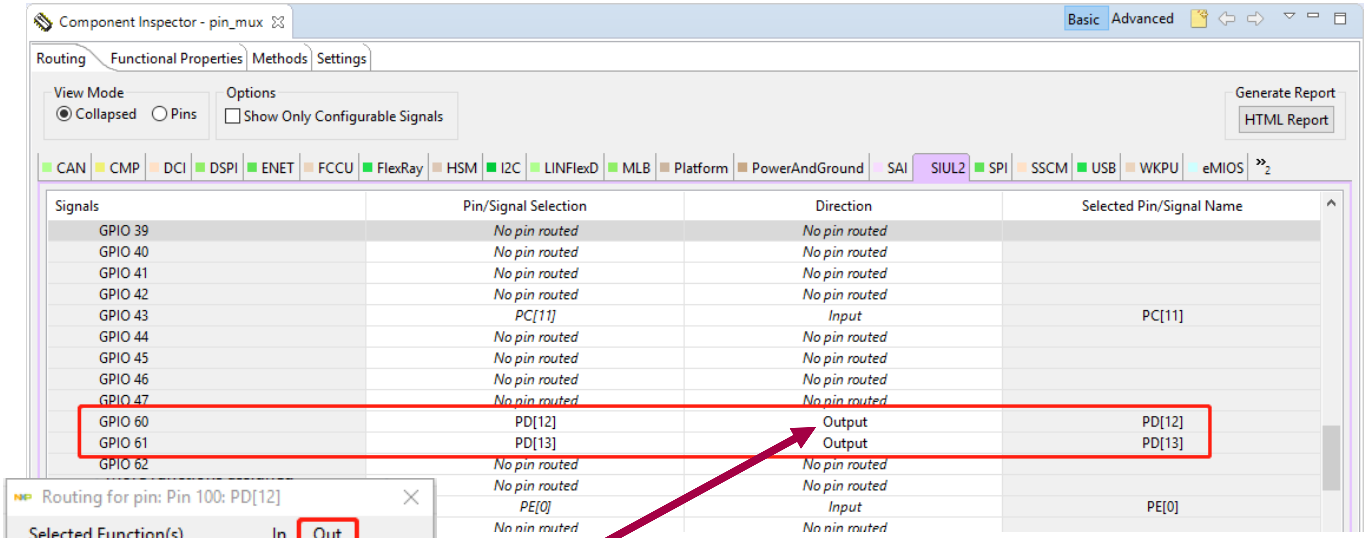


View Mode: ○ Collapsed ● Pins
Options: ☐ Show Only Configurable Signals
Pin Filter: pd

Pad Configuration

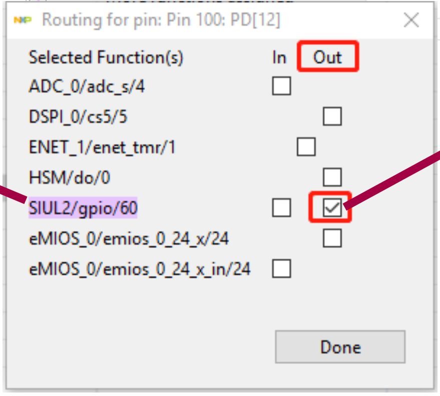
Pin	Pin Name	User Pin/Signal Name	Selected Function
77	PD[0]	PD[0]	More functions assigned
78	PD[1]	PD[1]	More functions assigned
79	PD[2]	PD[2]	More functions assigned
80	PD[3]	PD[3]	More functions assigned
81	PD[4]	PD[4]	More functions assigned
82	PD[5]	PD[5]	More functions assigned
83	PD[6]	PD[6]	More functions assigned
84	PD[7]	PD[7]	More functions assigned
87	PD[8]	PD[8]	More functions assigned
94	PD[9]	PD[9]	*none*
95	PD[10]	PD[10]	*none*
100	PD[12]	PD[12]	SIUL2/gpio/60
102	PD[13]	PD[13]	SIUL2/gpio/61
104	PD[14]	PD[14]	*none*
106	PD[15]	PD[15]	*none*





View Mode: ● Collapsed ○ Pins
Options: ☐ Show Only Configurable Signals
Generate Report

Signals	Pin/Signal Selection	Direction	Selected Pin/Signal Name
GPIO 39	*No pin routed*	*No pin routed*	
GPIO 40	*No pin routed*	*No pin routed*	
GPIO 41	*No pin routed*	*No pin routed*	
GPIO 42	*No pin routed*	*No pin routed*	
GPIO 43	*PC[11]*	*Input*	PC[11]
GPIO 44	*No pin routed*	*No pin routed*	
GPIO 45	*No pin routed*	*No pin routed*	
GPIO 46	*No pin routed*	*No pin routed*	
GPIO 47	*No pin routed*	*No pin routed*	
GPIO 60	PD[12]	Output	PD[12]
GPIO 61	PD[13]	Output	PD[13]
GPIO 62	*No pin routed*	*No pin routed*	
	No pin routed	*No pin routed*	
	PE[0]	*Input*	PE[0]
	No pin routed	*No pin routed*	

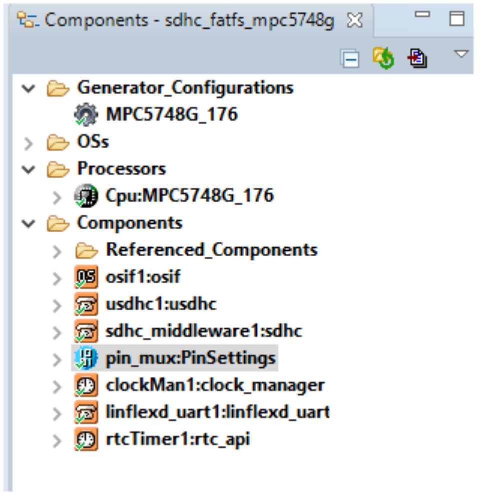


Selected Function(s)	In	Out
ADC_0/adc_s/4	☐	
DSPI_0/cs5/5		☐
ENET_1/enet_tmr/1	☐	
HSM/do/0		☐
SIUL2/gpio/60	☐	☑
eMIOS_0/emios_0_24_x/24		☐
eMIOS_0/emios_0_24_x_in/24	☐	

```c
void Peripheral_Power_Supply_Init(void)
{
    PINS_DRV_WritePin(PTB, 12, 0); //DoIP_Power enable

    PINS_DRV_WritePin(PTD, 12, 1);
    PINS_DRV_WritePin(PTD, 13, 1);
}
```

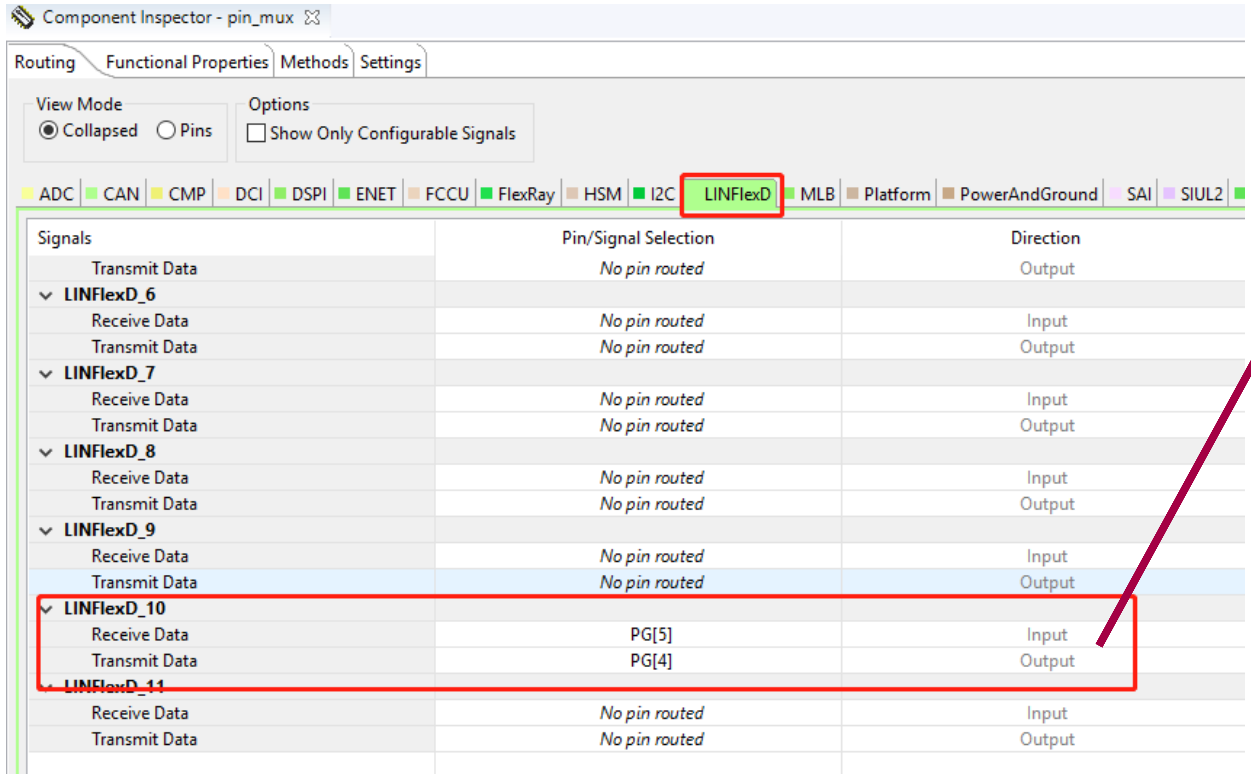
Hands-on – eMMC+Fatfs: UART configuration

Configure the port of UART



Components - sdhc_fatfs_mpc5748g

- Generator_Configurations
  - MPC5748G_176
- OSs
- Processors
  - Cpu:MPC5748G_176
- Components
  - Referenced_Components
  - osif1:osif
  - usdhc1:usdhc
  - sdhc_middleware1:sdhc
  - pin_mux:PinSettings
  - clockMan1:clock_manager
  - linflexd_uart1:linflexd_uart
  - rtcTimer1:rtc_api



Component Inspector - pin_mux

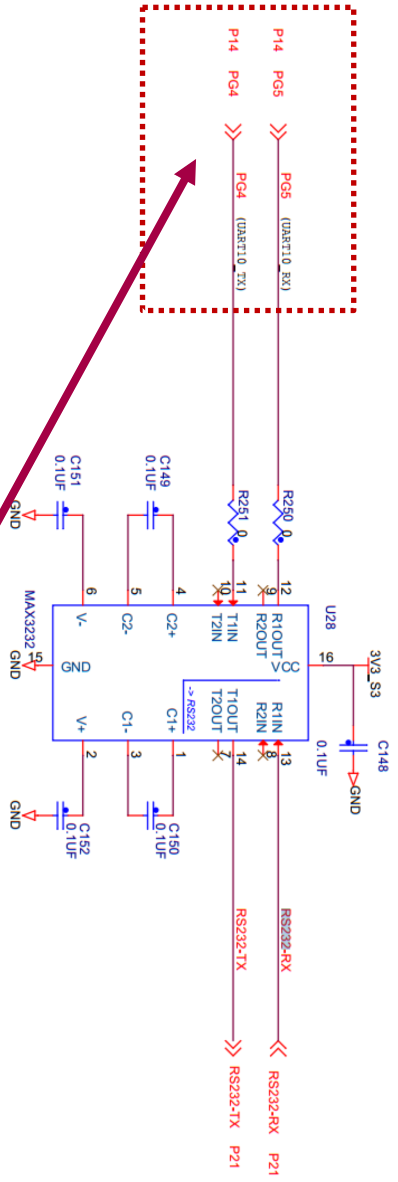
Routing | Functional Properties | Methods | Settings

View Mode: [✓ Collapsed] Pins

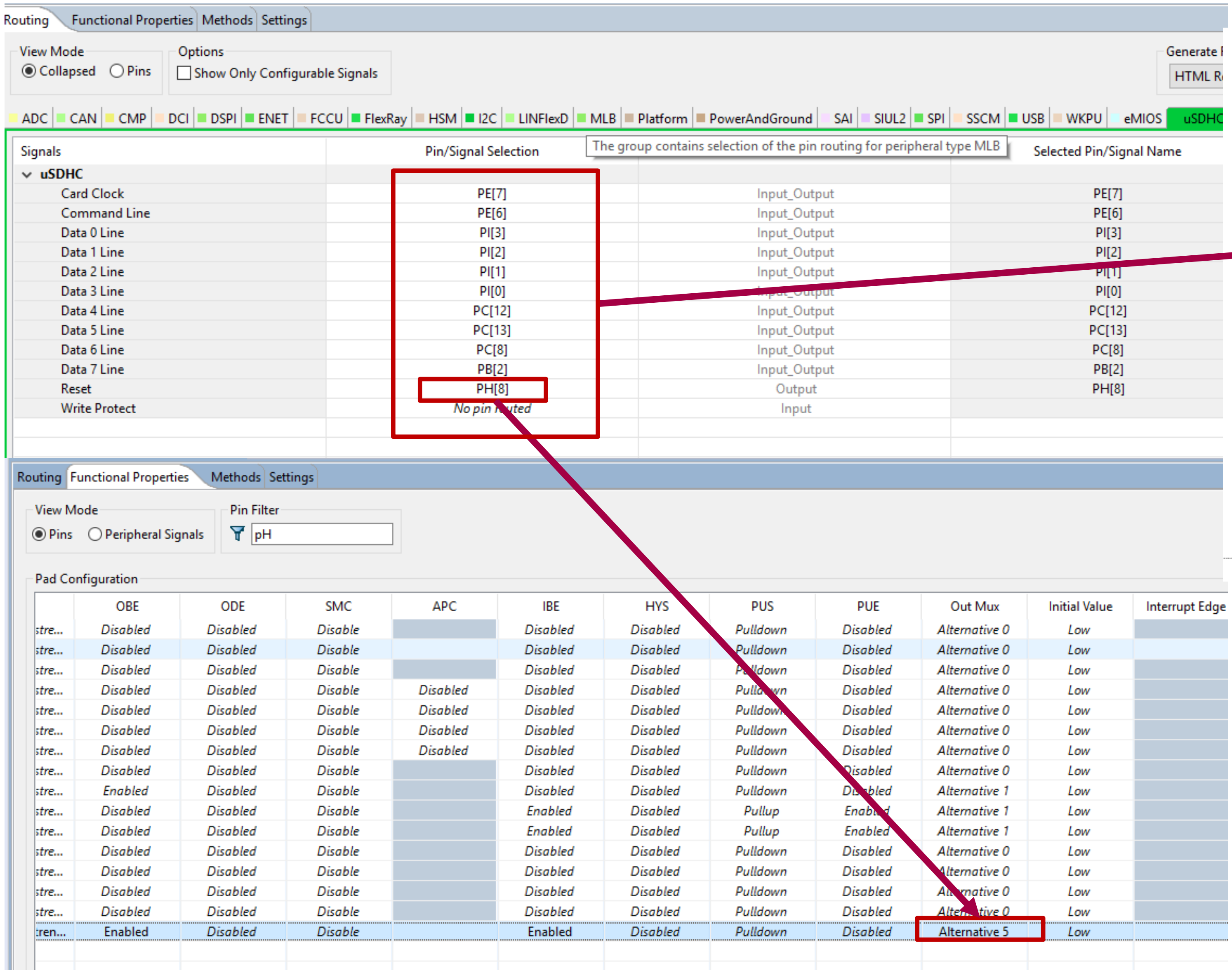
Options: ☐ Show Only Configurable Signals

ADC | CAN | CMP | DCI | DSPI | ENET | FCCU | FlexRay | HSM | I2C | LINFlexD | MLB | Platform | PowerAndGround | SAI | SIUL2

Signals	Pin/Signal Selection	Direction
Transmit Data	*No pin routed*	Output
LINFlexD_6		
Receive Data	*No pin routed*	Input
Transmit Data	*No pin routed*	Output
LINFlexD_7		
Receive Data	*No pin routed*	Input
Transmit Data	*No pin routed*	Output
LINFlexD_8		
Receive Data	*No pin routed*	Input
Transmit Data	*No pin routed*	Output
LINFlexD_9		
Receive Data	*No pin routed*	Input
Transmit Data	*No pin routed*	Output
LINFlexD_10		
Receive Data	PG[5]	Input
Transmit Data	PG[4]	Output
LINFlexD_11		
Receive Data	*No pin routed*	Input
Transmit Data	*No pin routed*	Output



Hands-on – eMMC+Fatfs: uSDHC configuration



Routing | Functional Properties | Methods | Settings

View Mode: [✓ Collapsed] Pins
Options: ☐ Show Only Configurable Signals
Generate HTML R

ADC | CAN | CMP | DCI | DSPI | ENET | FCCU | FlexRay | HSM | I2C | LINFlexD | MLB | Platform | PowerAndGround | SAI | SIUL2 | SPI | SSCM | USB | WKPU | eMIOS | uSDHC

The group contains selection of the pin routing for peripheral type MLB

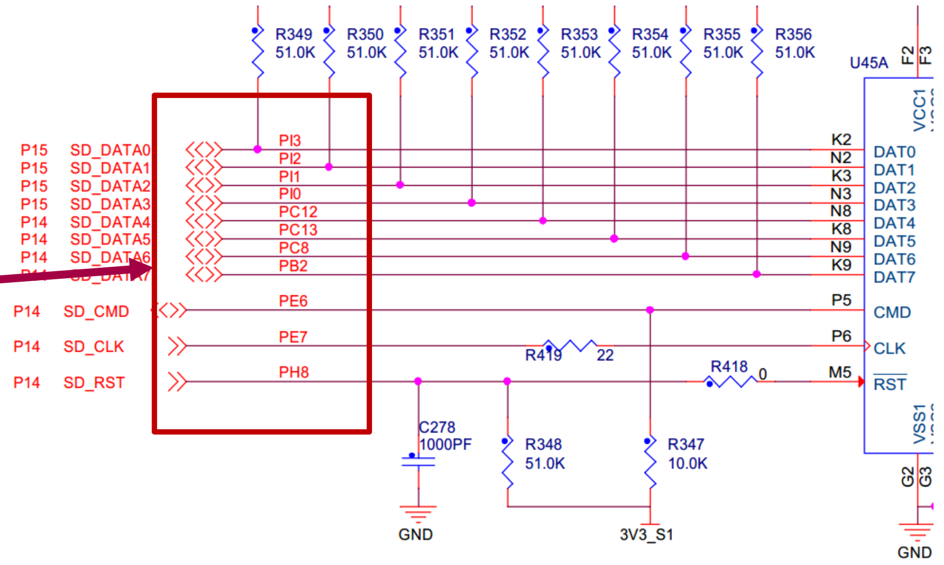
Signals	Pin/Signal Selection		Selected Pin/Signal Name
uSDHC			
Card Clock	PE[7]	Input_Output	PE[7]
Command Line	PE[6]	Input_Output	PE[6]
Data 0 Line	PI[3]	Input_Output	PI[3]
Data 1 Line	PI[2]	Input_Output	PI[2]
Data 2 Line	PI[1]	Input_Output	PI[1]
Data 3 Line	PI[0]	Input_Output	PI[0]
Data 4 Line	PC[12]	Input_Output	PC[12]
Data 5 Line	PC[13]	Input_Output	PC[13]
Data 6 Line	PC[8]	Input_Output	PC[8]
Data 7 Line	PB[2]	Input_Output	PB[2]
Reset	PH[8]	Output	PH[8]
Write Protect	No pin routed	Input	

Routing | Functional Properties | Methods | Settings

View Mode: [✓ Pins] Peripheral Signals
Pin Filter: pH

Pad Configuration

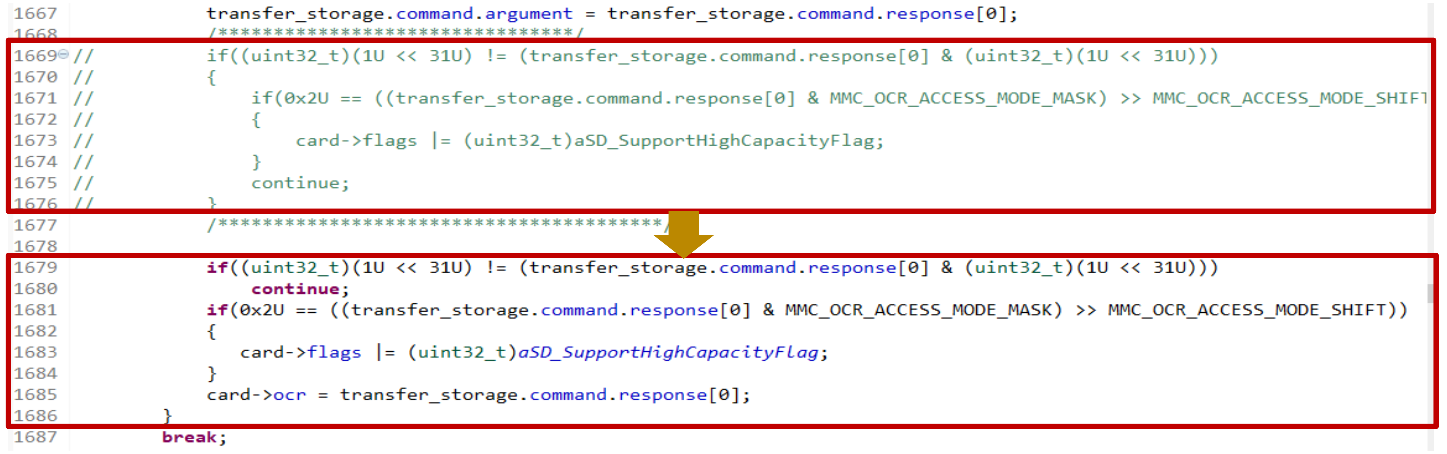
	OBE	ODE	SMC	APC	IBE	HYS	PUS	PUE	Out Mux	Initial Value	Interrupt Edge
stre...	Disabled	Disabled	Disable		Disabled	Disabled	Pulldown	Disabled	Alternative 0	Low	
stre...	Disabled	Disabled	Disable		Disabled	Disabled	Pulldown	Disabled	Alternative 0	Low	
stre...	Disabled	Disabled	Disable		Disabled	Disabled	Pulldown	Disabled	Alternative 0	Low	
stre...	Disabled	Disabled	Disable	Disabled	Disabled	Disabled	Pulldown	Disabled	Alternative 0	Low	
stre...	Disabled	Disabled	Disable	Disabled	Disabled	Disabled	Pulldown	Disabled	Alternative 0	Low	
stre...	Disabled	Disabled	Disable	Disabled	Disabled	Disabled	Pulldown	Disabled	Alternative 0	Low	
stre...	Disabled	Disabled	Disable	Disabled	Disabled	Disabled	Pulldown	Disabled	Alternative 0	Low	
stre...	Disabled	Disabled	Disable		Disabled	Disabled	Pulldown	Disabled	Alternative 0	Low	
stre...	Enabled	Disabled	Disable		Disabled	Disabled	Pulldown	Disabled	Alternative 1	Low	
stre...	Disabled	Disabled	Disable		Enabled	Disabled	Pullup	Enabled	Alternative 1	Low	
stre...	Disabled	Disabled	Disable		Enabled	Disabled	Pullup	Enabled	Alternative 1	Low	
stre...	Disabled	Disabled	Disable		Disabled	Disabled	Pulldown	Disabled	Alternative 0	Low	
stre...	Disabled	Disabled	Disable		Disabled	Disabled	Pulldown	Disabled	Alternative 0	Low	
stre...	Disabled	Disabled	Disable		Disabled	Disabled	Pulldown	Disabled	Alternative 0	Low	
stre...	Disabled	Disabled	Disable		Disabled	Disabled	Pulldown	Disabled	Alternative 0	Low	
tren...	Enabled	Disabled	Disable		Enabled	Disabled	Pulldown	Disabled	Alternative 5	Low	



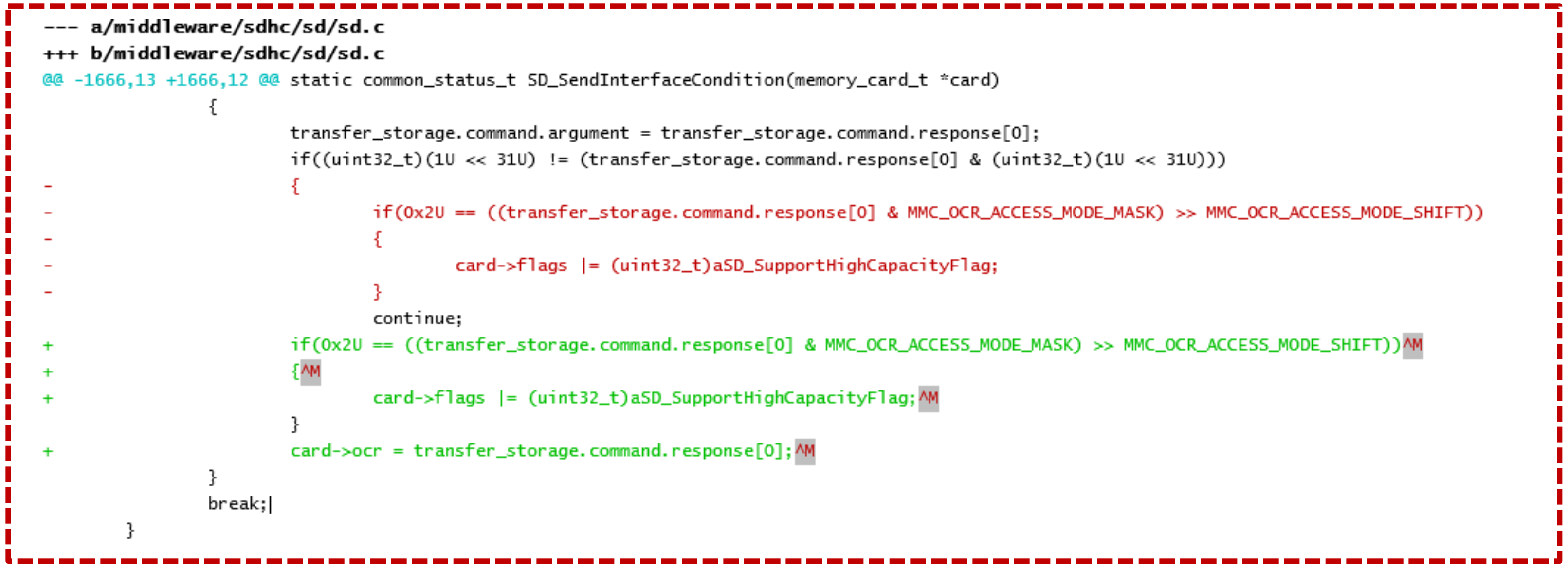
Hands-on – eMMC+Fatfs: Application Code

(1) Fix a problem that exists in SDK

- {Project Name} -> SDK -> middleware -> sdhc -> sd-> sd.c

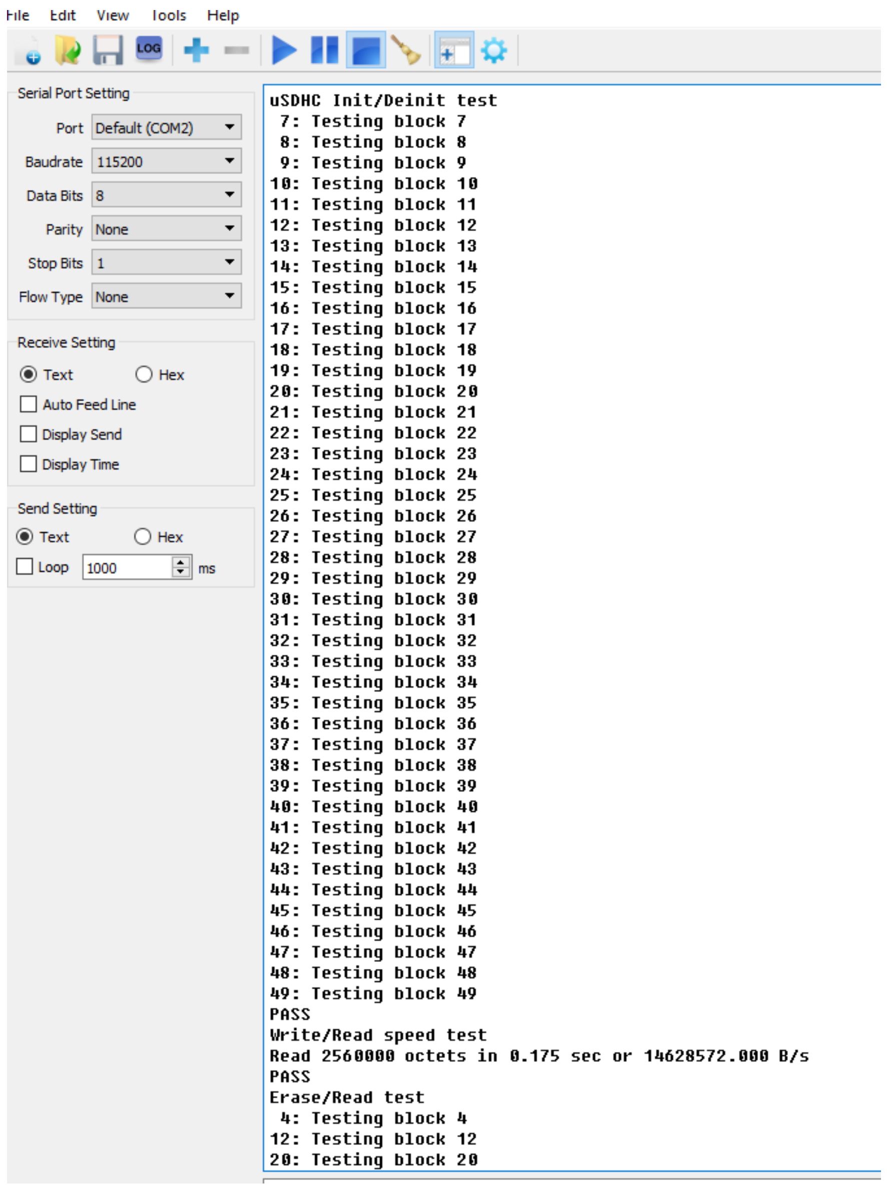


```
            transfer_storage.command.argument = transfer_storage.command.response[0];
            /********************************/
//          if((uint32_t)(1U << 31U) != (transfer_storage.command.response[0] & (uint32_t)(1U << 31U)))
//          {
//              if(0x2U == ((transfer_storage.command.response[0] & MMC_OCR_ACCESS_MODE_MASK) >> MMC_OCR_ACCESS_MODE_SHIF
//              {
//                  card->flags |= (uint32_t)aSD_SupportHighCapacityFlag;
//              }
//              continue;
//          }
            /****************************************/

            if((uint32_t)(1U << 31U) != (transfer_storage.command.response[0] & (uint32_t)(1U << 31U)))
                continue;
            if(0x2U == ((transfer_storage.command.response[0] & MMC_OCR_ACCESS_MODE_MASK) >> MMC_OCR_ACCESS_MODE_SHIFT))
            {
                card->flags |= (uint32_t)aSD_SupportHighCapacityFlag;
            }
            card->ocr = transfer_storage.command.response[0];
        }
        break;
```



```
--- a/middleware/sdhc/sd/sd.c
+++ b/middleware/sdhc/sd/sd.c
@@ -1666,13 +1666,12 @@ static common_status_t SD_SendInterfaceCondition(memory_card_t *card)
                {
                        transfer_storage.command.argument = transfer_storage.command.response[0];
                        if((uint32_t)(1U << 31U) != (transfer_storage.command.response[0] & (uint32_t)(1U << 31U)))
-                       {
-                               if(0x2U == ((transfer_storage.command.response[0] & MMC_OCR_ACCESS_MODE_MASK) >> MMC_OCR_ACCESS_MODE_SHIFT))
-                               {
-                                       card->flags |= (uint32_t)aSD_SupportHighCapacityFlag;
-                               }
                                continue;
+                       if(0x2U == ((transfer_storage.command.response[0] & MMC_OCR_ACCESS_MODE_MASK) >> MMC_OCR_ACCESS_MODE_SHIFT))^M
+                       {^M
+                               card->flags |= (uint32_t)aSD_SupportHighCapacityFlag;^M
                        }
+                       card->ocr = transfer_storage.command.response[0];^M
                }
                break;|
        }
```

Hands-on – eMMC+Fatfs: Build and Debug

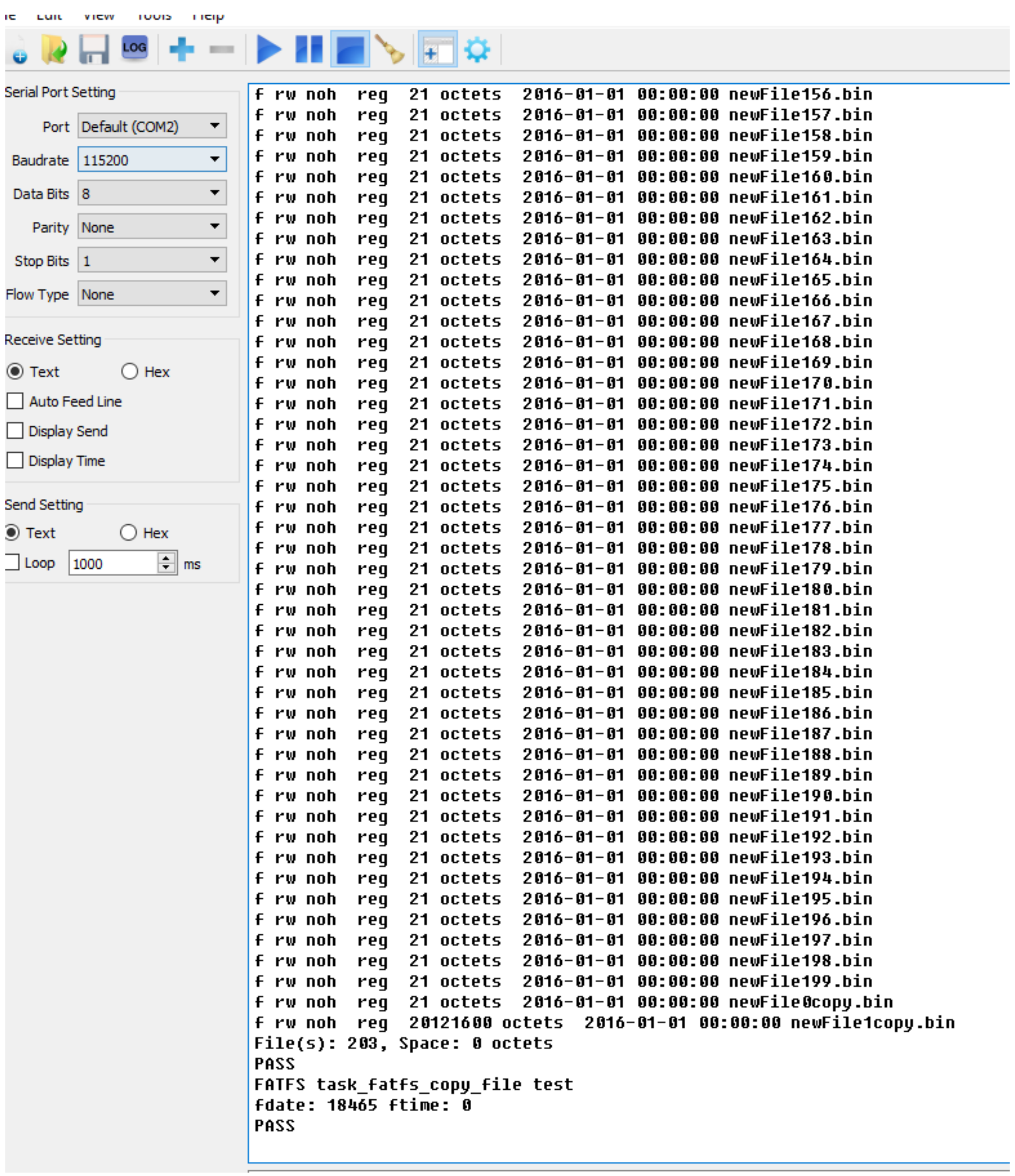


Port: Default (COM2)
Baudrate: 115200
Data Bits: 8
Parity: None
Stop Bits: 1
Flow Type: None

Receive Setting: [✓ Text] Hex
☐ Auto Feed Line
☐ Display Send
☐ Display Time

Send Setting: [✓ Text] Hex
☐ Loop 1000 ms

```
uSDHC Init/Deinit test
 7: Testing block 7
 8: Testing block 8
 9: Testing block 9
10: Testing block 10
11: Testing block 11
12: Testing block 12
13: Testing block 13
14: Testing block 14
15: Testing block 15
16: Testing block 16
17: Testing block 17
18: Testing block 18
19: Testing block 19
20: Testing block 20
21: Testing block 21
22: Testing block 22
23: Testing block 23
24: Testing block 24
25: Testing block 25
26: Testing block 26
27: Testing block 27
28: Testing block 28
29: Testing block 29
30: Testing block 30
31: Testing block 31
32: Testing block 32
33: Testing block 33
34: Testing block 34
35: Testing block 35
36: Testing block 36
37: Testing block 37
38: Testing block 38
39: Testing block 39
40: Testing block 40
41: Testing block 41
42: Testing block 42
43: Testing block 43
44: Testing block 44
45: Testing block 45
46: Testing block 46
47: Testing block 47
48: Testing block 48
49: Testing block 49
PASS
Write/Read speed test
Read 2560000 octets in 0.175 sec or 14628572.000 B/s
PASS
Erase/Read test
 4: Testing block 4
12: Testing block 12
20: Testing block 20
```



Port: Default (COM2)
Baudrate: 115200
Data Bits: 8
Parity: None
Stop Bits: 1
Flow Type: None

Receive Setting: [✓ Text] Hex
☐ Auto Feed Line
☐ Display Send
☐ Display Time

Send Setting: [✓ Text] Hex
☐ Loop 1000 ms

```
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile156.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile157.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile158.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile159.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile160.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile161.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile162.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile163.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile164.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile165.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile166.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile167.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile168.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile169.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile170.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile171.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile172.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile173.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile174.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile175.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile176.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile177.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile178.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile179.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile180.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile181.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile182.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile183.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile184.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile185.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile186.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile187.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile188.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile189.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile190.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile191.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile192.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile193.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile194.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile195.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile196.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile197.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile198.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile199.bin
f rw noh  reg  21 octets  2016-01-01 00:00:00 newFile0copy.bin
f rw noh  reg  20121600 octets  2016-01-01 00:00:00 newFile1copy.bin
File(s): 203, Space: 0 octets
PASS
FATFS task_fatfs_copy_file test
fdate: 18465 ftime: 0
PASS
```

SECURE CONNECTIONS
FOR A SMARTER WORLD