GLDBOX REAL TIME DRIVER EXAMPLE ENABLEMENT GUIDE

SECURE CONNECTIONS
FOR A SMARTER WORLD

PUBLIC

Contents

Hands on UART Real Time Driver example

Hands on ETH Real Time Driver example

Hands on CAN Real Time Driver example

Hardware Requirement and Software Installation

Hardware Requirement

- S32G-VNP-RDB2

- S32 Debug Probe

- AD/DC power supply

- Serial port cable for UART example

Software Installation

- S32DS3.4 according to S32G-VNP-GLDBOX Software Enablement Guide
- SW32_RTD_4.4_1.0.0(RTD) according to S32G-VNP-GLDBOX Software Enablement Guide

01.

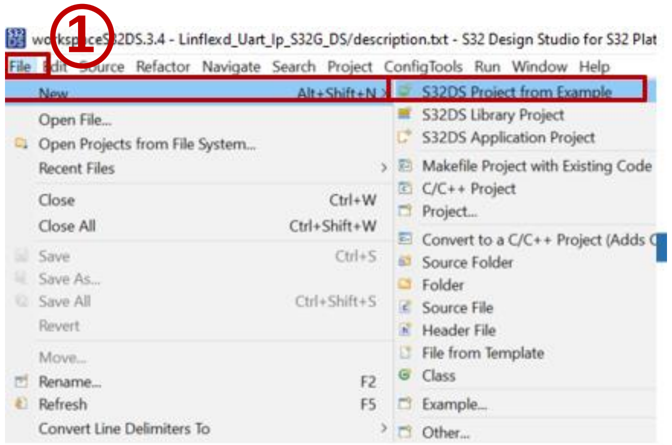
Hands on UART Example

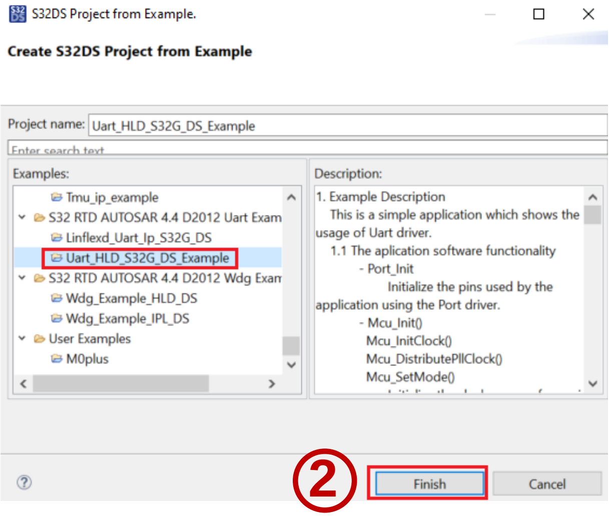
Hands on UART: Objective

- How to import the UART example into S32DS
- How to configure the clock of UART via S32DS
- How to configure the UART setting via S32DS
- How to debug the UART example with S32 debug probe

Hands on UART: Import UART example project

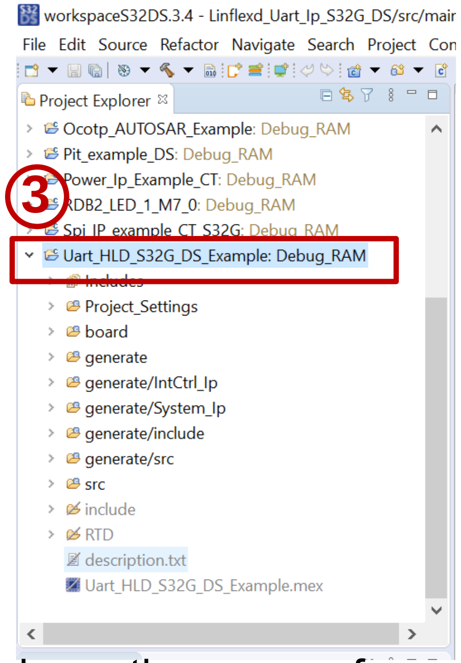
Open S32DS3.4, go to “File -> New -> S32DS Project From Example”. Select “**Uart_HLD_S32G_DS_Example**” example, Then click on “Finish”. The project should now be copied into current workspace.


①
workspaceS32DS.3.4 - Linflexd_Uart_Ip_S32G_DS/description.txt - S32 Design Studio for S32 Plat
File Edit Source Refactor Navigate Search Project ConfigTools Run Window Help
New Alt+Shift+N
Open File...
Open Projects from File System...
Recent Files
Close Ctrl+W
Close All Ctrl+Shift+W
Save Ctrl+S
Save As...
Save All Ctrl+Shift+S
Revert
Move...
Rename... F2
Refresh F5
Convert Line Delimiters To
S32DS Project from Example
S32DS Library Project
S32DS Application Project
Makefile Project with Existing Code
C/C++ Project
Project...
Convert to a C/C++ Project (Adds C
Source Folder
Folder
Source File
Header File
File from Template
Class
Example...
Other...

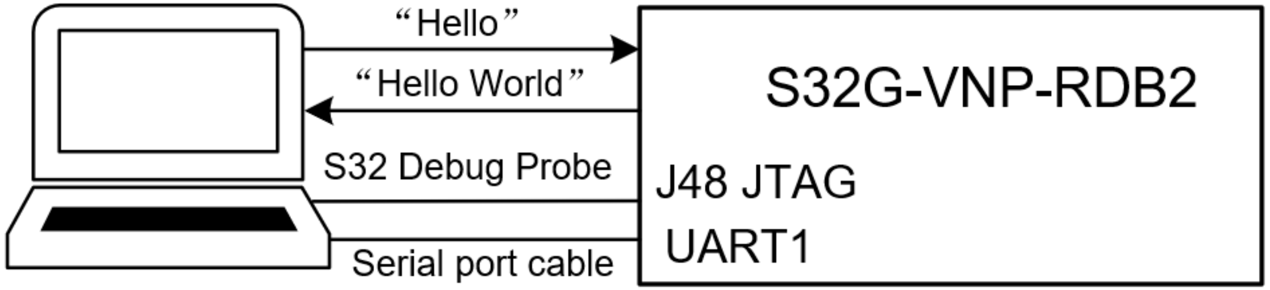


Project name: Uart_HLD_S32G_DS_Example

1. Example Description
This is a simple application which shows the usage of Uart driver.
1.1 The aplication software functionality
- Port_Init
Initialize the pins used by the application using the Port driver.
- Mcu_Init()
Mcu_InitClock()
Mcu_DistributePllClock()
Mcu_SetMode()


workspaceS32DS.3.4 - Linflexd_Uart_Ip_S32G_DS/src/mair
File Edit Source Refactor Navigate Search Project Con
Project Explorer
Ocotp_AUTOSAR_Example: Debug_RAM
Pit_example_DS: Debug_RAM
③
Power_Ip_Example_CT: Debug_RAM
RDB2_LED_1_M7_0: Debug_RAM
Spi_IP_example_CT_S32G: Debug_RAM
Uart_HLD_S32G_DS_Example: Debug_RAM
Includes
Project_Settings
board
generate
generate/IntCtrl_Ip
generate/System_Ip
generate/include
generate/src
src
include
RTD
description.txt
Uart_HLD_S32G_DS_Example.mex

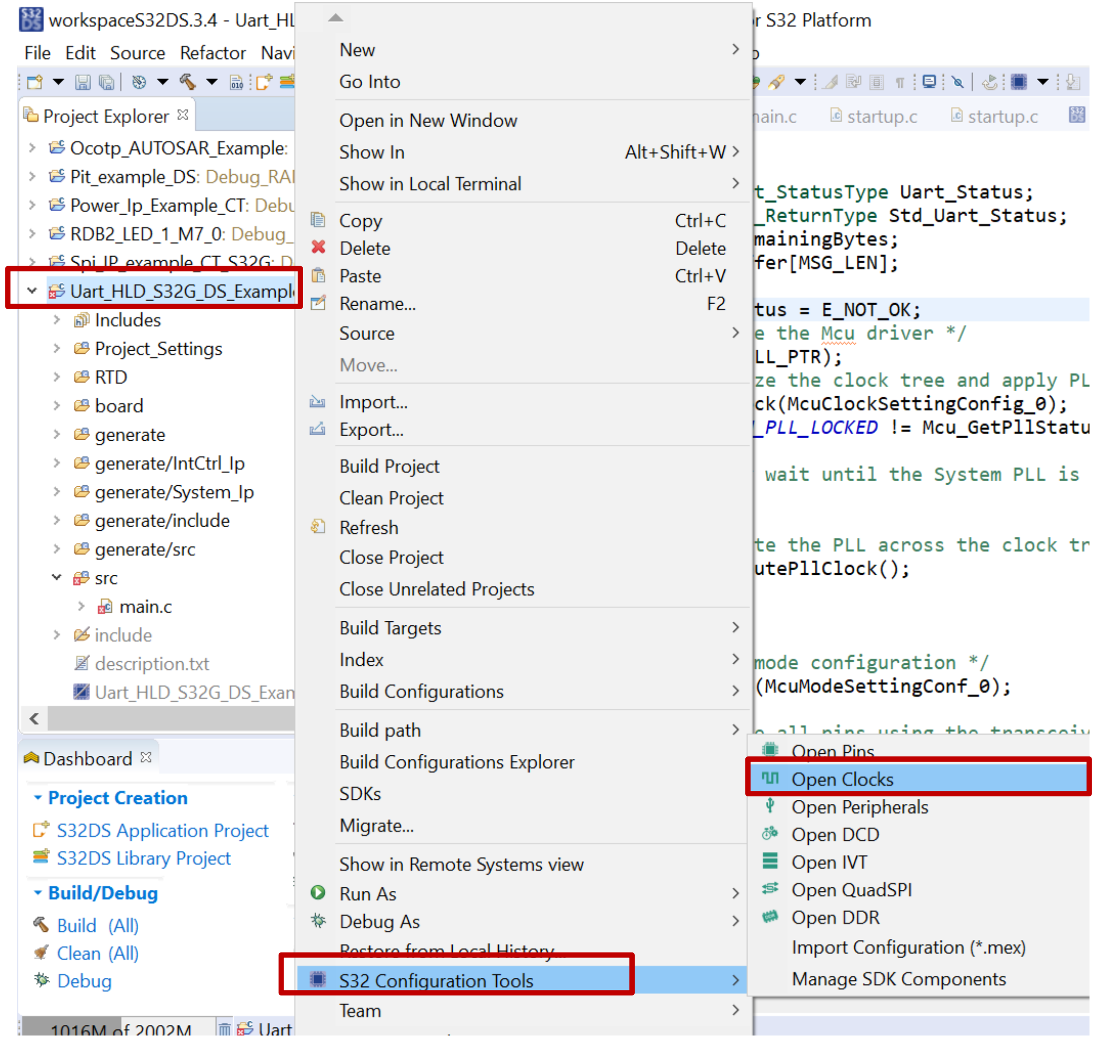
The purpose of “Uart_HLD_S32G_DS_Example” example is a simple application which shows the usage of UART driver.



Hands on UART: Clock Configuration 1

Go to desired configuration tool:

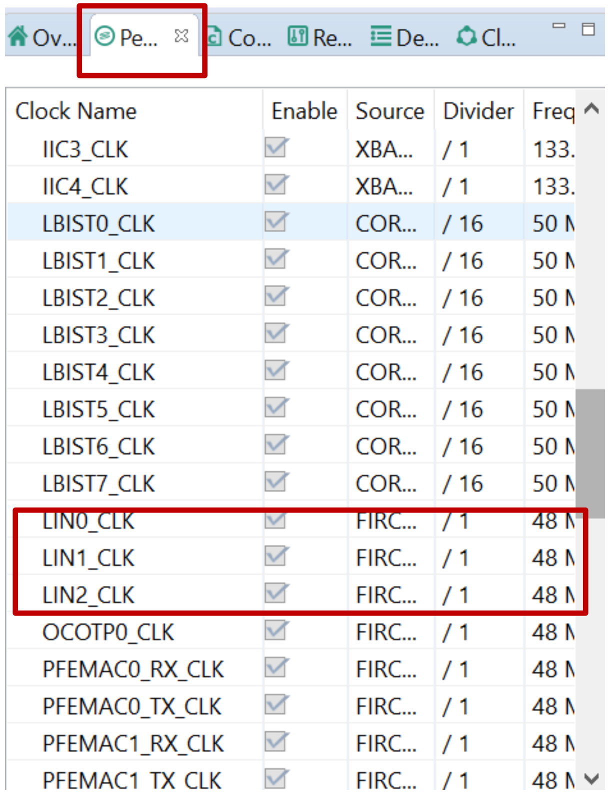
- Right click on Project,
- Select S32 Configuration Tool…
- Select Open Clocks



```
t_StatusType Uart_Status;
_ReturnType Std_Uart_Status;
mainingBytes;
fer[MSG_LEN];

tus = E_NOT_OK;
e the Mcu driver */
LL_PTR);
ze the clock tree and apply PL
ck(McuClockSettingConfig_0);
_PLL_LOCKED != Mcu_GetPllStatu

 wait until the System PLL is

te the PLL across the clock tr
utePllClock();

mode configuration */
(McuModeSettingConf_0);
```

Hands on UART: Clock Configuration 2

Open the **Peripheral Clock View**, Double click the Lin module. The **Clocks Diagram** will show the power tree .In Uart_HLD_S32G_DS_Example project. The default clock configuration of UART is 48 MHZ which comes from FIRC directly



Clock Name	Enable	Source	Divider	Freq
IIC3_CLK	☑	XBA...	/ 1	133.
IIC4_CLK	☑	XBA...	/ 1	133.
LBIST0_CLK	☑	COR...	/ 16	50 M
LBIST1_CLK	☑	COR...	/ 16	50 M
LBIST2_CLK	☑	COR...	/ 16	50 M
LBIST3_CLK	☑	COR...	/ 16	50 M
LBIST4_CLK	☑	COR...	/ 16	50 M
LBIST5_CLK	☑	COR...	/ 16	50 M
LBIST6_CLK	☑	COR...	/ 16	50 M
LBIST7_CLK	☑	COR...	/ 16	50 M
LIN0_CLK	☑	FIRC...	/ 1	48 M
LIN1_CLK	☑	FIRC...	/ 1	48 M
LIN2_CLK	☑	FIRC...	/ 1	48 M
OCOTP0_CLK	☑	FIRC...	/ 1	48 M
PFEMAC0_RX_CLK	☑	FIRC...	/ 1	48 M
PFEMAC0_TX_CLK	☑	FIRC...	/ 1	48 M
PFEMAC1_RX_CLK	☑	FIRC...	/ 1	48 M
PFEMAC1_TX_CLK	☑	FIRC...	/ 1	48 M

Fig. Peripheral Clock View

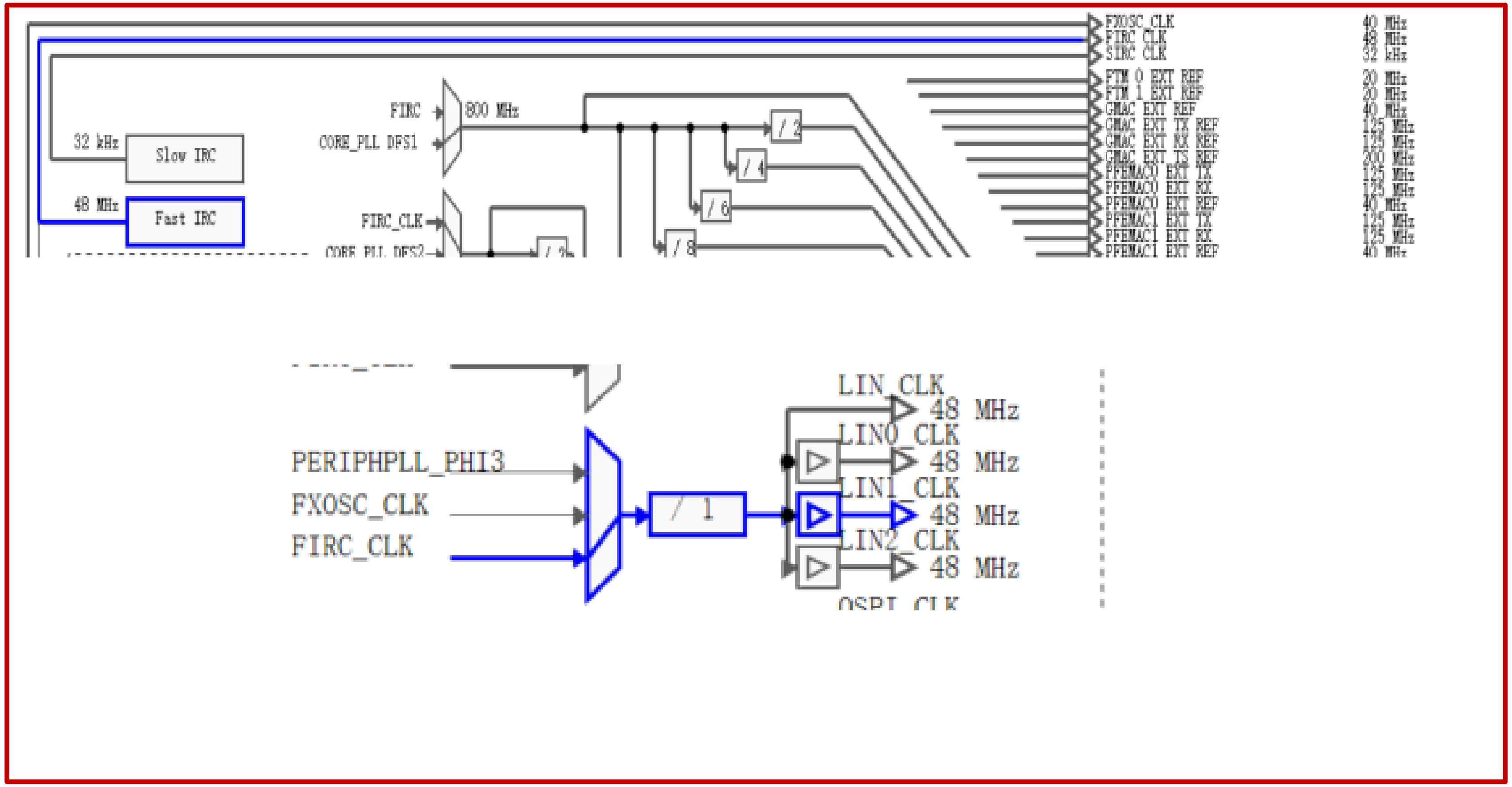
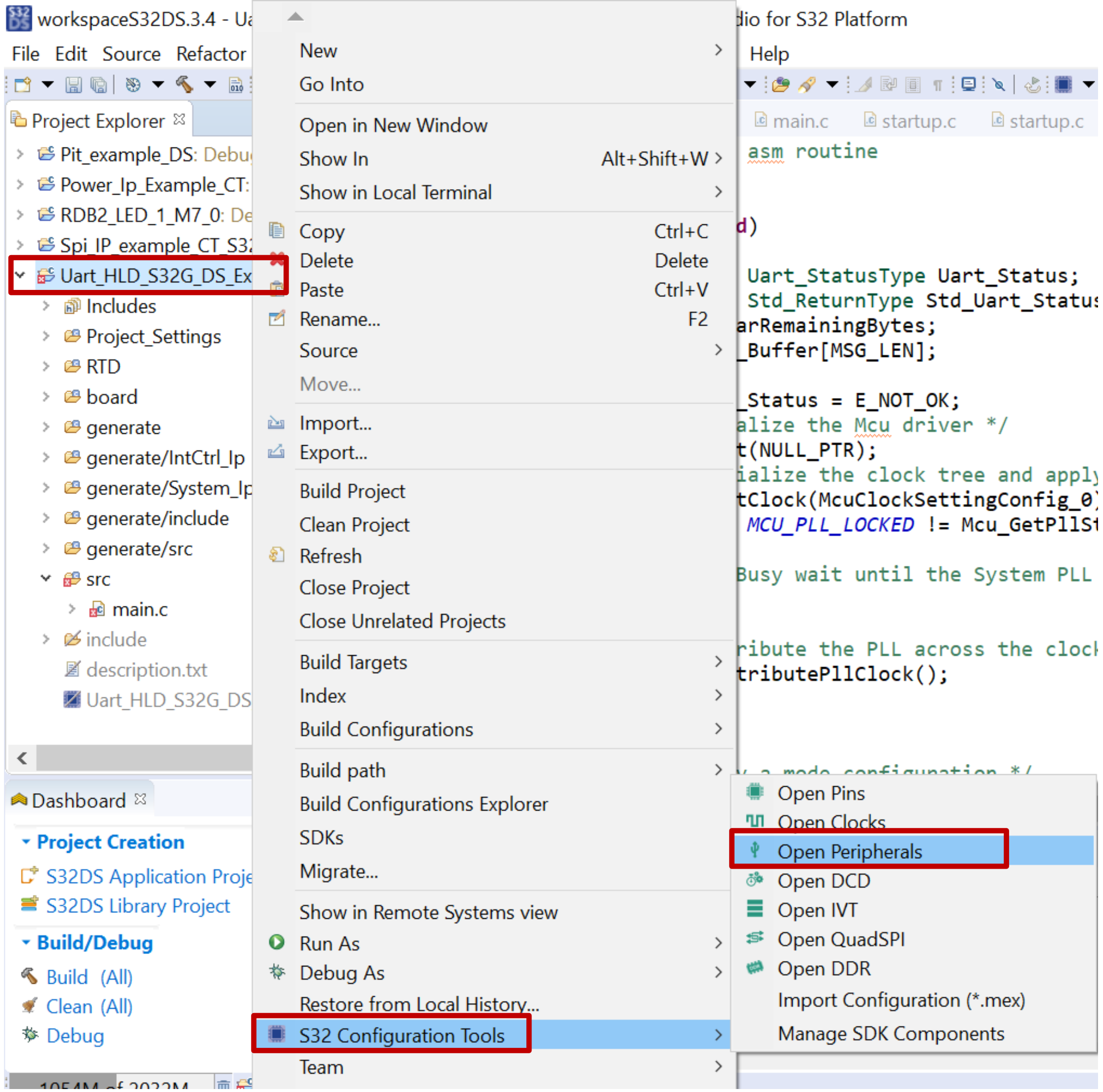


Fig. Clocks Diagram

Hands on UART: UART Configuration 1

Open the Clocks Diagram:

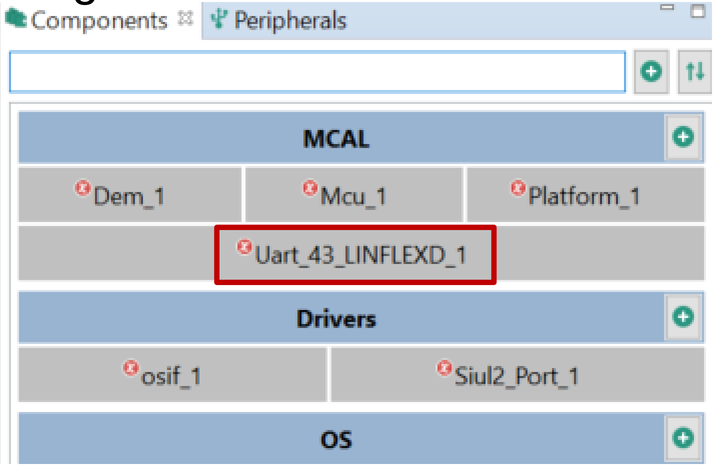
- Right click on Project,
- Select S32 Configuration Tool…
- Select Peripherals



```
asm routine

d)

 Uart_StatusType Uart_Status;
 Std_ReturnType Std_Uart_Statu
arRemainingBytes;
_Buffer[MSG_LEN];

_Status = E_NOT_OK;
alize the Mcu driver */
t(NULL_PTR);
ialize the clock tree and apply
tClock(McuClockSettingConfig_0)
 MCU_PLL_LOCKED != Mcu_GetPllSt

Busy wait until the System PLL

ribute the PLL across the clock
tributePllClock();
```

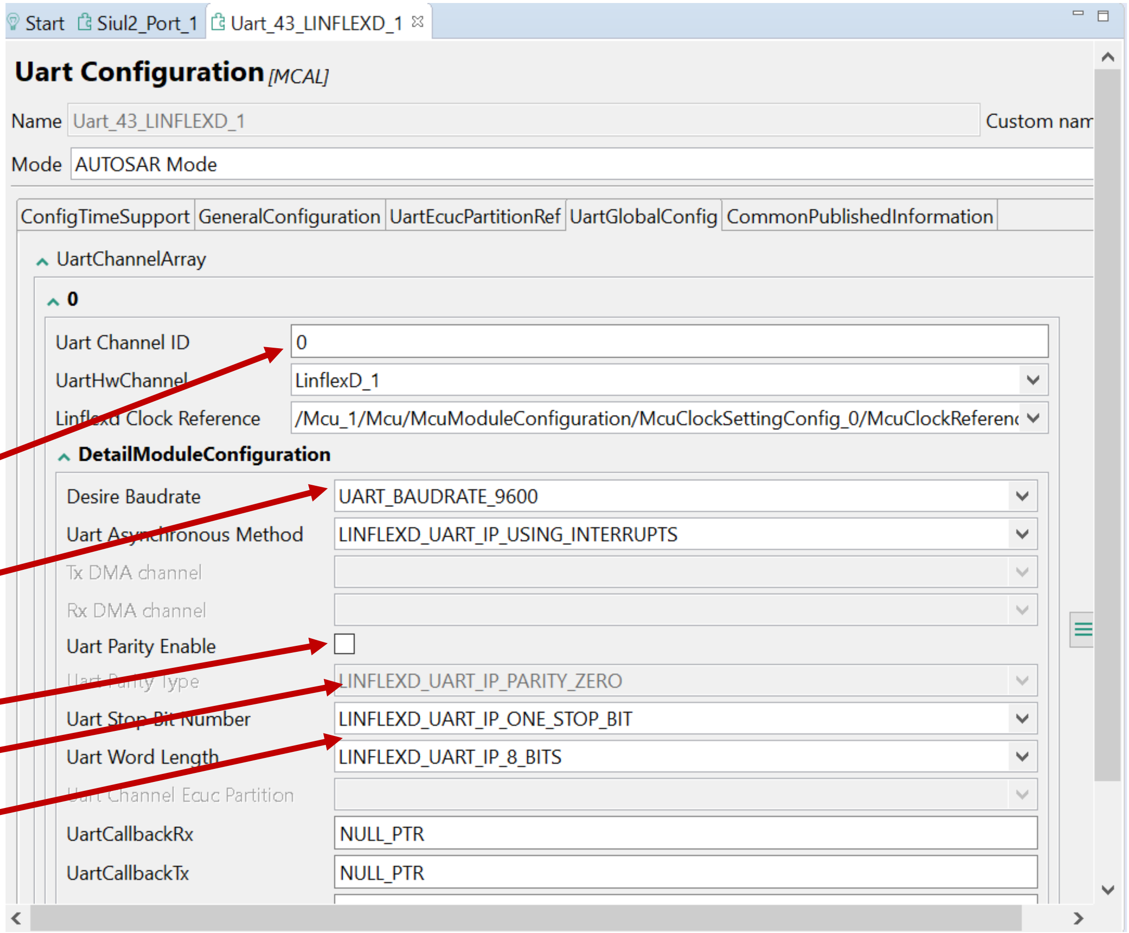
Hands on UART: UART Configuration 2

The **Components** shows all drivers which used by this example, the **UART_43_LINFLEXD_1** includes the configuration of UART driver



UART default configuration:

- - Select correct COM Port
- - Select Baudrate of 9600
- - Select none parity checking
- - Select 1 stop bits
- - Select 8 data bits



Uart Configuration *[MCAL]*

Name: Uart_43_LINFLEXD_1 Custom nam

Mode: AUTOSAR Mode

UartChannelArray

0

Uart Channel ID: 0

UartHwChannel: LinflexD_1

Linflexd Clock Reference: /Mcu_1/Mcu/McuModuleConfiguration/McuClockSettingConfig_0/McuClockReferenc

DetailModuleConfiguration

Desire Baudrate: UART_BAUDRATE_9600

Uart Asynchronous Method: LINFLEXD_UART_IP_USING_INTERRUPTS

Tx DMA channel:

Rx DMA channel:

Uart Parity Enable: □

Uart Parity Type: LINFLEXD_UART_IP_PARITY_ZERO

Uart Stop Bit Number: LINFLEXD_UART_IP_ONE_STOP_BIT

Uart Word Length: LINFLEXD_UART_IP_8_BITS

Uart Channel Ecuc Partition:

UartCallbackRx: NULL_PTR

UartCallbackTx: NULL_PTR

Hands on UART: Update code

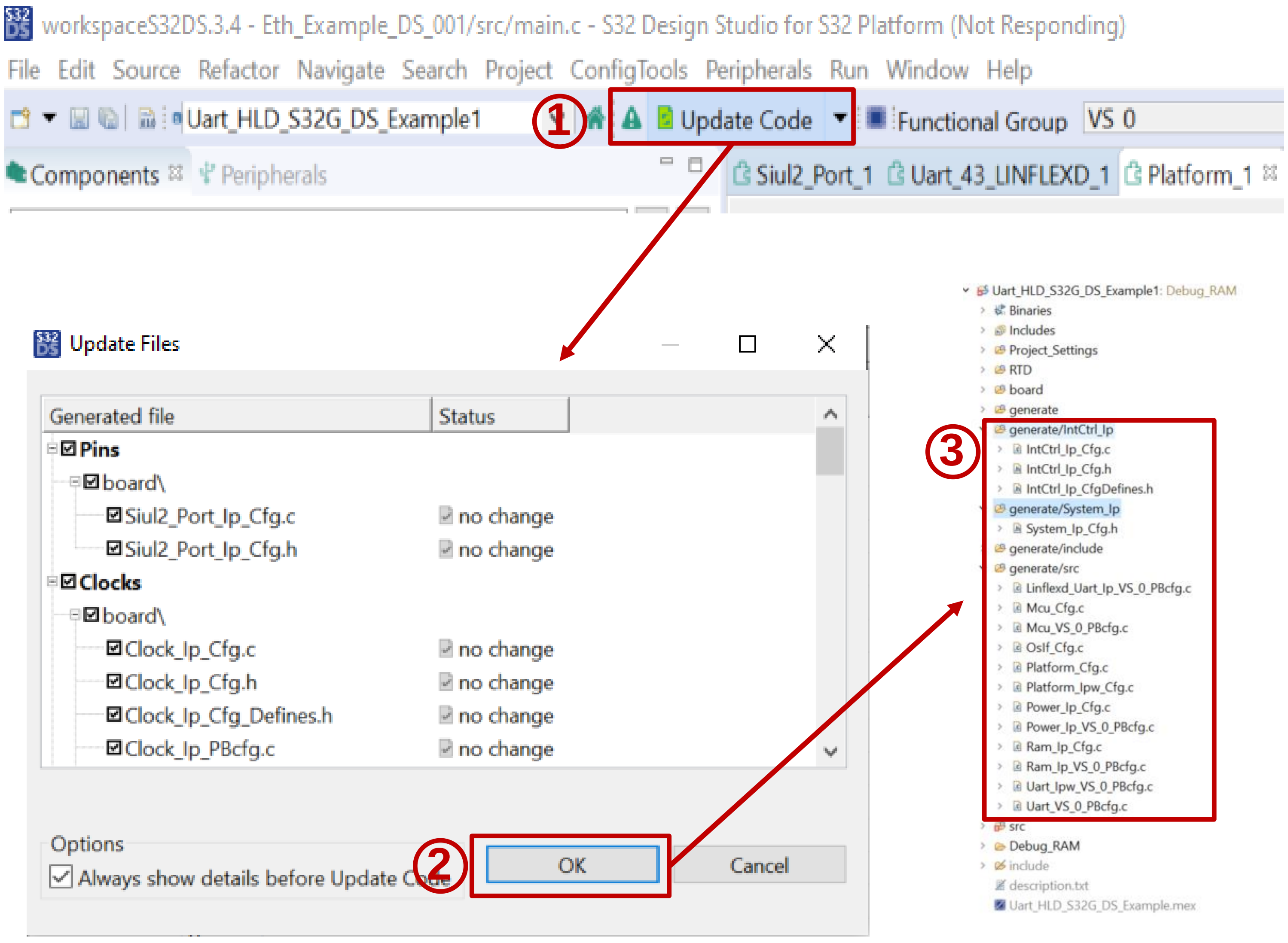
Generate code method:

1.Click on any configuration tool, like Pins

Then click **Update Code** (ensure desired project is selected!)

2.The Update Files window pops up. It shows the detail update information. Click **ok** button.

3.The configuration .c and .h file will be generated at “generate” folder.



workspaceS32DS.3.4 - Eth_Example_DS_001/src/main.c - S32 Design Studio for S32 Platform (Not Responding)

Generated file	Status
☑ Pins	
☑ board\	
☑ Siul2_Port_Ip_Cfg.c	no change
☑ Siul2_Port_Ip_Cfg.h	no change
☑ Clocks	
☑ board\	
☑ Clock_Ip_Cfg.c	no change
☑ Clock_Ip_Cfg.h	no change
☑ Clock_Ip_Cfg_Defines.h	no change
☑ Clock_Ip_PBcfg.c	no change

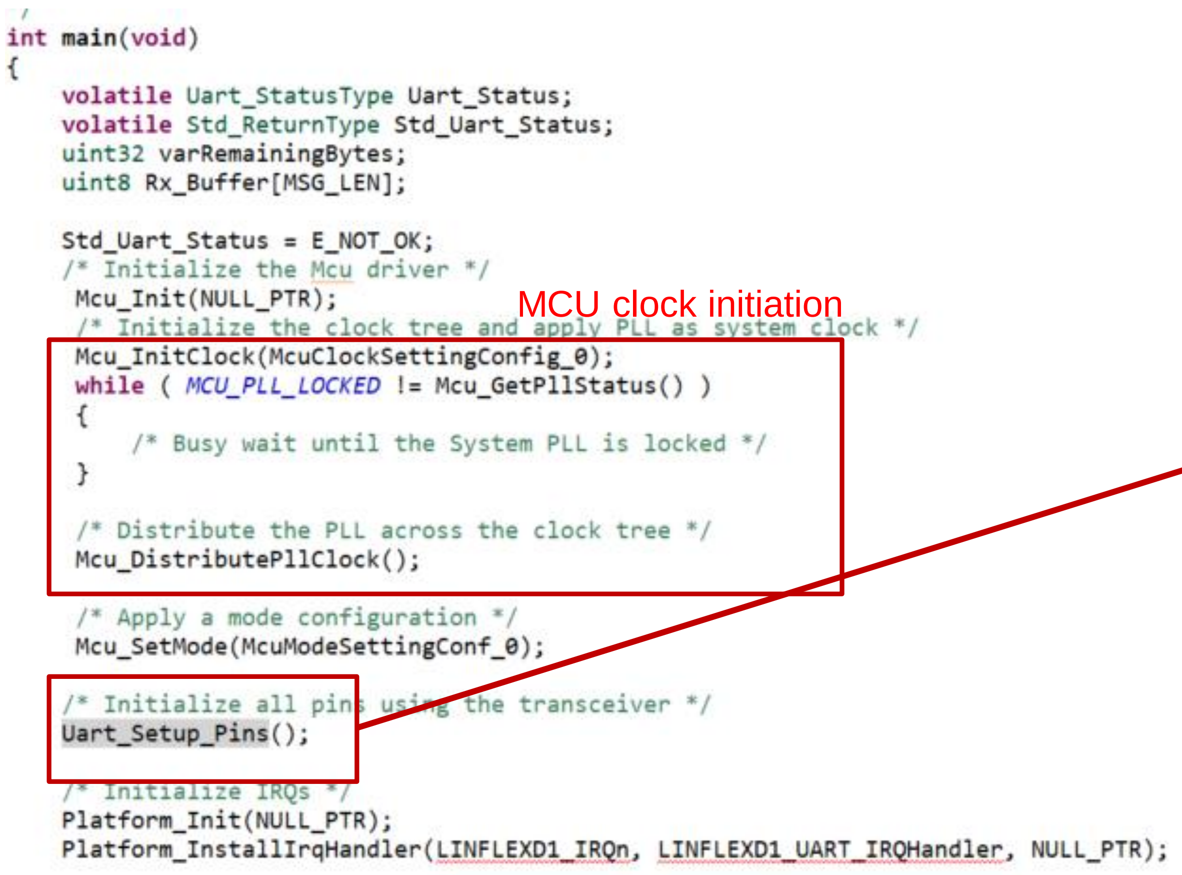
Options

☑ Always show details before Update Code

```
Uart_HLD_S32G_DS_Example1: Debug_RAM
  Binaries
  Includes
  Project_Settings
  RTD
  board
  generate
  generate/IntCtrl_Ip
    IntCtrl_Ip_Cfg.c
    IntCtrl_Ip_Cfg.h
    IntCtrl_Ip_CfgDefines.h
  generate/System_Ip
    System_Ip_Cfg.h
  generate/include
  generate/src
    Linflexd_Uart_Ip_VS_0_PBcfg.c
    Mcu_Cfg.c
    Mcu_VS_0_PBcfg.c
    OsIf_Cfg.c
    Platform_Cfg.c
    Platform_Ipw_Cfg.c
    Power_Ip_Cfg.c
    Power_Ip_VS_0_PBcfg.c
    Ram_Ip_Cfg.c
    Ram_Ip_VS_0_PBcfg.c
    Uart_Ipw_VS_0_PBcfg.c
    Uart_VS_0_PBcfg.c
  src
  Debug_RAM
  include
  description.txt
  Uart_HLD_S32G_DS_Example.mex
```

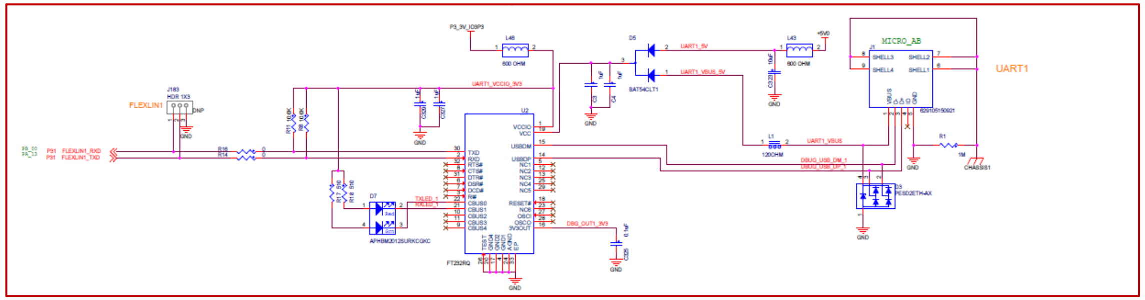
Hands on UART: Application code 1

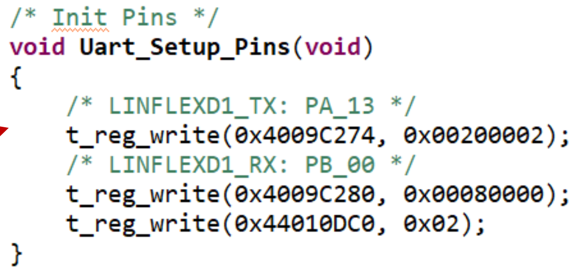
Open the main.c file in S32DS



```
int main(void)
{
    volatile Uart_StatusType Uart_Status;
    volatile Std_ReturnType Std_Uart_Status;
    uint32 varRemainingBytes;
    uint8 Rx_Buffer[MSG_LEN];

    Std_Uart_Status = E_NOT_OK;
    /* Initialize the Mcu driver */
     Mcu_Init(NULL_PTR);
     /* Initialize the clock tree and apply PLL as system clock */
     Mcu_InitClock(McuClockSettingConfig_0);
     while ( MCU_PLL_LOCKED != Mcu_GetPllStatus() )
     {
         /* Busy wait until the System PLL is locked */
     }

     /* Distribute the PLL across the clock tree */
     Mcu_DistributePllClock();

     /* Apply a mode configuration */
     Mcu_SetMode(McuModeSettingConf_0);

    /* Initialize all pins using the transceiver */
    Uart_Setup_Pins();

    /* Initialize IRQs */
    Platform_Init(NULL_PTR);
    Platform_InstallIrqHandler(LINFLEXD1_IRQn, LINFLEXD1_UART_IRQHandler, NULL_PTR);
```

MCU clock initiation

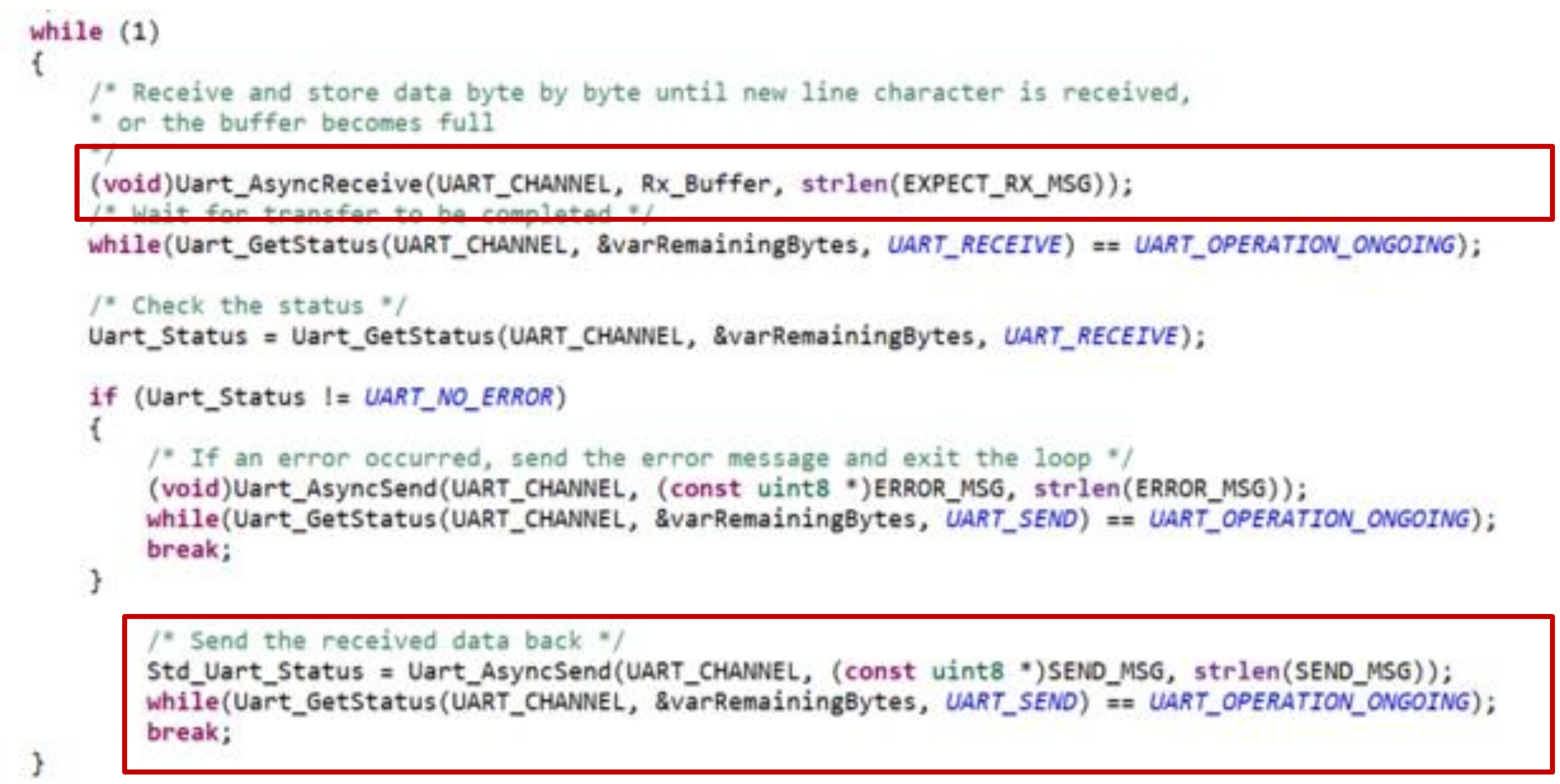




```
/* Init Pins */
void Uart_Setup_Pins(void)
{
    /* LINFLEXD1_TX: PA_13 */
    t_reg_write(0x4009C274, 0x00200002);
    /* LINFLEXD1_RX: PB_00 */
    t_reg_write(0x4009C280, 0x00080000);
    t_reg_write(0x44010DC0, 0x02);
}
```

In Uart_HLD_S32G_DS_Example Project. Initialization of pins is writing register directly.

Hands on UART: Application code 2



```
while (1)
{
    /* Receive and store data byte by byte until new line character is received,
    * or the buffer becomes full
    */
    (void)Uart_AsyncReceive(UART_CHANNEL, Rx_Buffer, strlen(EXPECT_RX_MSG));
    /* Wait for transfer to be completed */
    while(Uart_GetStatus(UART_CHANNEL, &varRemainingBytes, UART_RECEIVE) == UART_OPERATION_ONGOING);

    /* Check the status */
    Uart_Status = Uart_GetStatus(UART_CHANNEL, &varRemainingBytes, UART_RECEIVE);

    if (Uart_Status != UART_NO_ERROR)
    {
        /* If an error occurred, send the error message and exit the loop */
        (void)Uart_AsyncSend(UART_CHANNEL, (const uint8 *)ERROR_MSG, strlen(ERROR_MSG));
        while(Uart_GetStatus(UART_CHANNEL, &varRemainingBytes, UART_SEND) == UART_OPERATION_ONGOING);
        break;
    }

        /* Send the received data back */
        Std_Uart_Status = Uart_AsyncSend(UART_CHANNEL, (const uint8 *)SEND_MSG, strlen(SEND_MSG));
        while(Uart_GetStatus(UART_CHANNEL, &varRemainingBytes, UART_SEND) == UART_OPERATION_ONGOING);
        break;
}
```

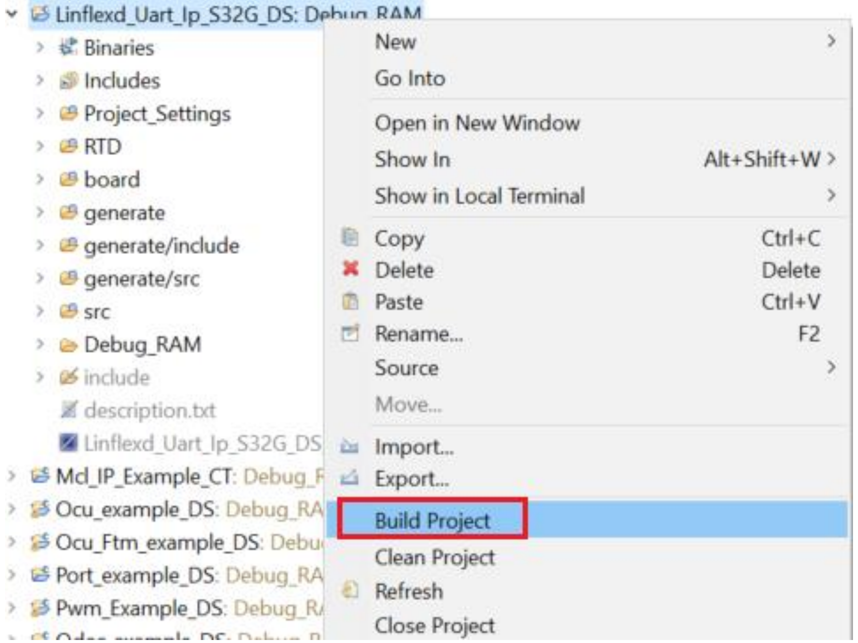
Receive data from user

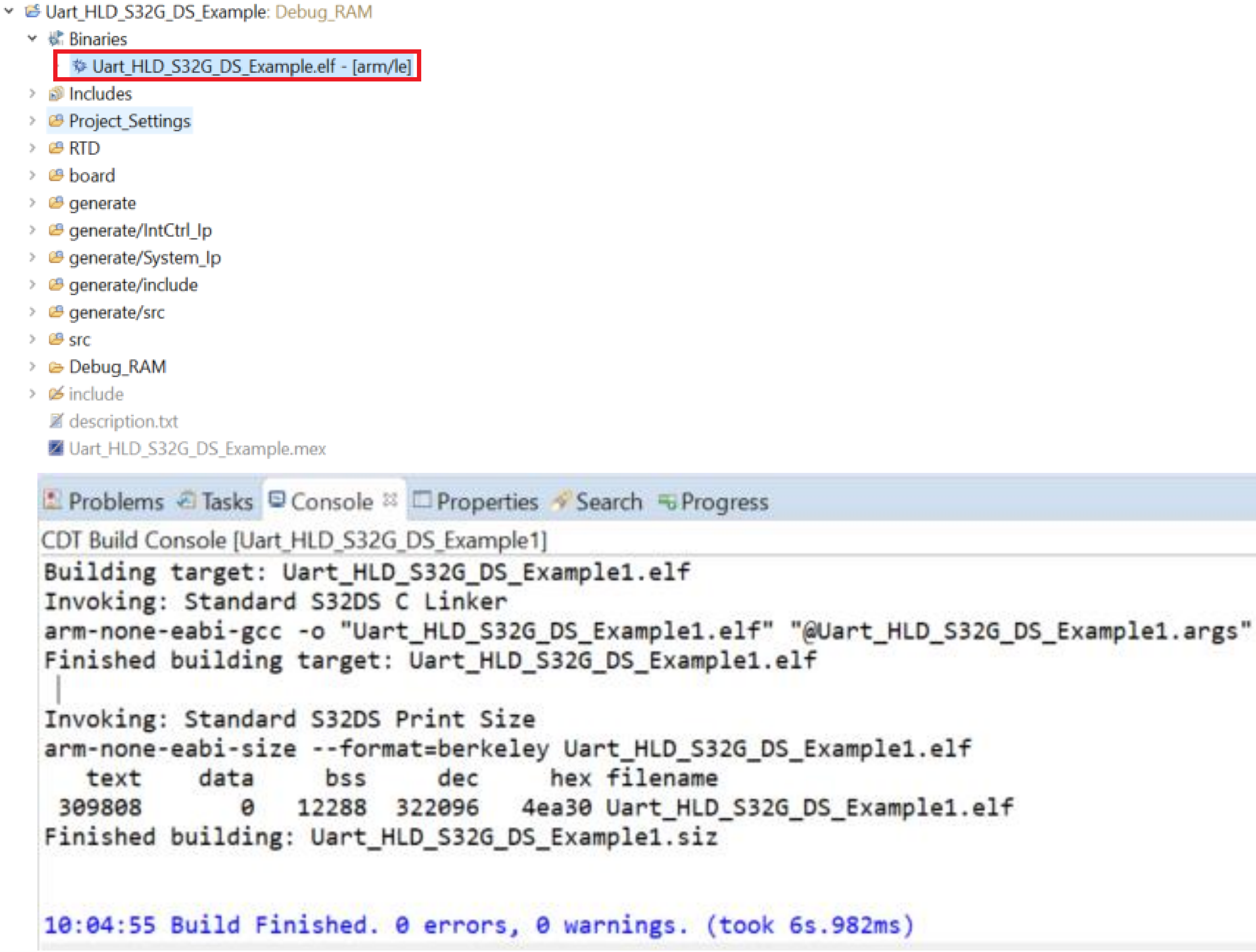
Echo the received data back

Hands on UART: Build and Debug 1

Build the target :

- Right click on Project,
- Select Build Project
- Print Build information on Console window
- Uart_HLD_S32G_DS_Example1.elf is generated





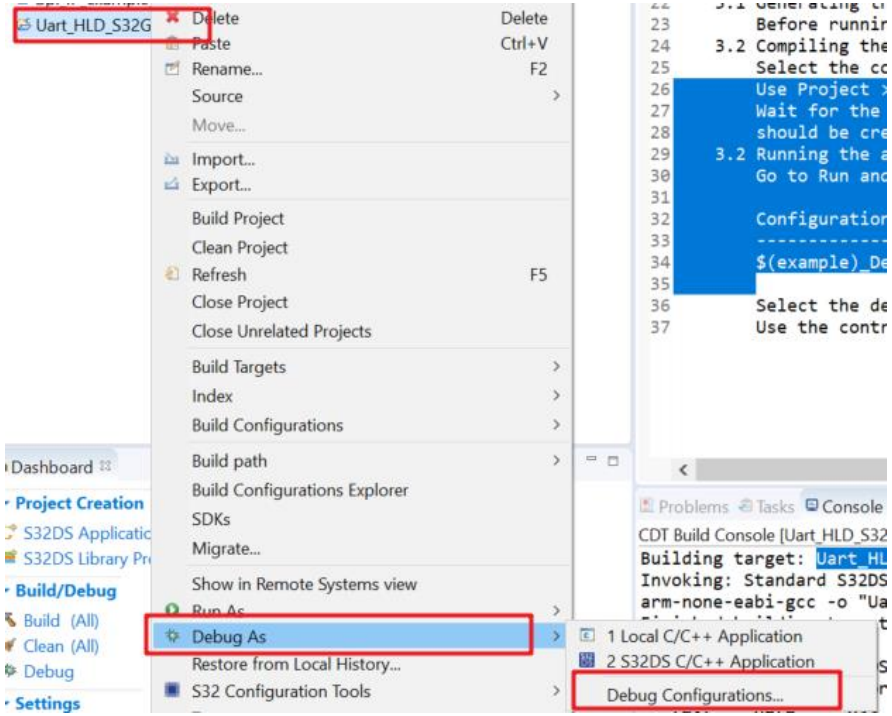
CDT Build Console [Uart_HLD_S32G_DS_Example1]

```
Building target: Uart_HLD_S32G_DS_Example1.elf
Invoking: Standard S32DS C Linker
arm-none-eabi-gcc -o "Uart_HLD_S32G_DS_Example1.elf" "@Uart_HLD_S32G_DS_Example1.args"
Finished building target: Uart_HLD_S32G_DS_Example1.elf

Invoking: Standard S32DS Print Size
arm-none-eabi-size --format=berkeley Uart_HLD_S32G_DS_Example1.elf
   text    data     bss     dec     hex filename
 309808       0   12288  322096   4ea30 Uart_HLD_S32G_DS_Example1.elf
Finished building: Uart_HLD_S32G_DS_Example1.siz


10:04:55 Build Finished. 0 errors, 0 warnings. (took 6s.982ms)
```

Hands on UART: Build and Debug 2

Go to debug configuration:

- Right click on Project,
- Select the Debug As
- Click Configurations

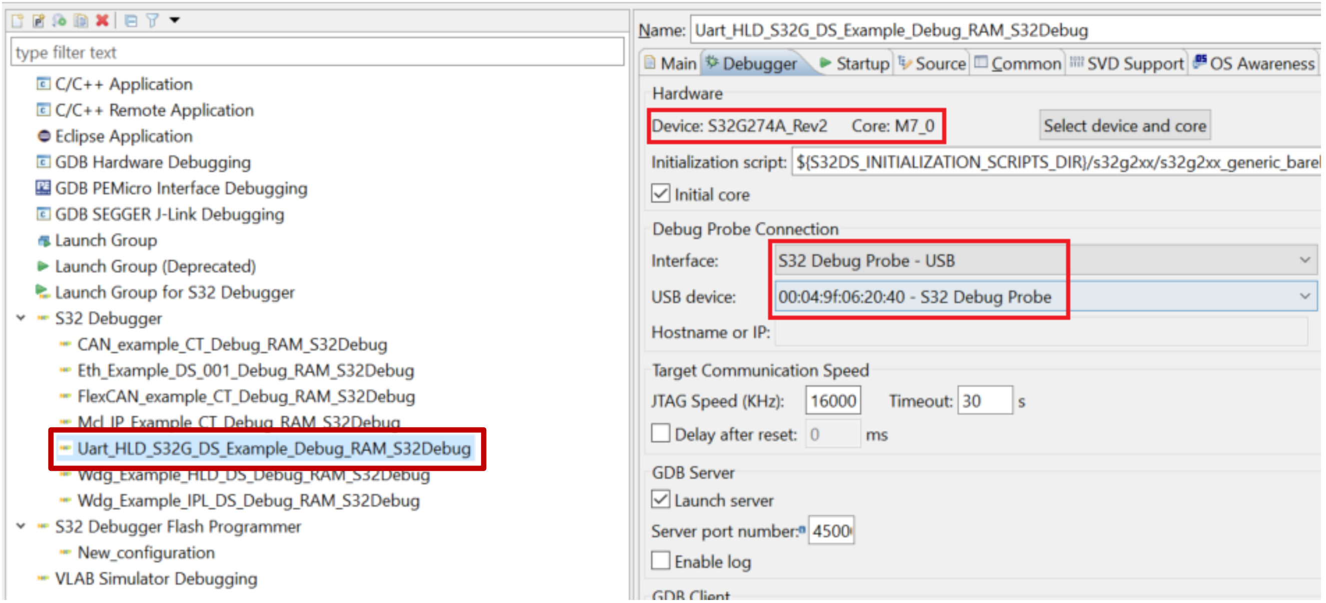


```
        Before runnir
    3.2 Compiling the
        Select the cc
        Use Project >
        Wait for the
        should be cre
    3.2 Running the a
        Go to Run anc

        Configuratior
        -------------
        $(example)_De

        Select the de
        Use the contr
```

```
CDT Build Console [Uart_HLD_S32
Building target: Uart_HL
Invoking: Standard S32DS
arm-none-eabi-gcc -o "Ua
```

Debug configuration set:

- Click target project ,
- Select the target device
- Select target S32 Debug Probe


type filter text
C/C++ Application
C/C++ Remote Application
Eclipse Application
GDB Hardware Debugging
GDB PEMicro Interface Debugging
GDB SEGGER J-Link Debugging
Launch Group
Launch Group (Deprecated)
Launch Group for S32 Debugger
S32 Debugger
CAN_example_CT_Debug_RAM_S32Debug
Eth_Example_DS_001_Debug_RAM_S32Debug
FlexCAN_example_CT_Debug_RAM_S32Debug
Mcl_IP_Example_CT_Debug_RAM_S32Debug
Uart_HLD_S32G_DS_Example_Debug_RAM_S32Debug
Wdg_Example_HLD_DS_Debug_RAM_S32Debug
Wdg_Example_IPL_DS_Debug_RAM_S32Debug
S32 Debugger Flash Programmer
New_configuration
VLAB Simulator Debugging
Main
Debugger
Startup
Source
Common
SVD Support
OS Awareness
Select device and core

Name: Uart_HLD_S32G_DS_Example_Debug_RAM_S32Debug

Hardware

Device: S32G274A_Rev2 Core: M7_0

Initialization script: ${S32DS_INITIALIZATION_SCRIPTS_DIR}/s32g2xx/s32g2xx_generic_barel

☑ Initial core

Debug Probe Connection

Interface: S32 Debug Probe - USB

USB device: 00:04:9f:06:20:40 - S32 Debug Probe

Hostname or IP:

Target Communication Speed

JTAG Speed (KHz): 16000 Timeout: 30 s

☐ Delay after reset: 0 ms

GDB Server

☑ Launch server

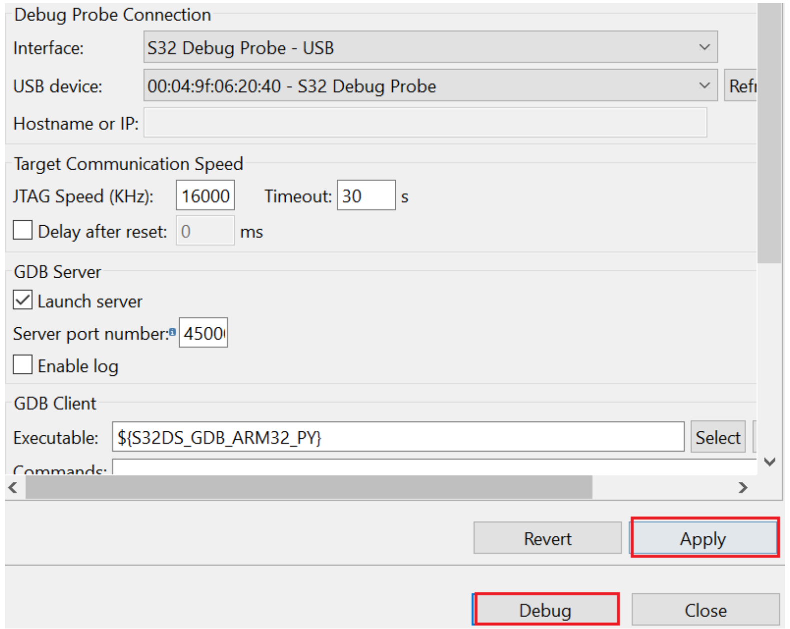
Server port number: 4500

☐ Enable log

GDB Client

Hands on UART : Debug and run

Click on “Apply”, then click on “Debug”. the perspective will jump to the Debug Perspective, and you can use the controls to control the program flow.



Debug Probe Connection

Interface: S32 Debug Probe - USB

USB device: 00:04:9f:06:20:40 - S32 Debug Probe

Hostname or IP:

Target Communication Speed

JTAG Speed (KHz): 16000 Timeout: 30 s

□ Delay after reset: 0 ms

GDB Server

☑ Launch server

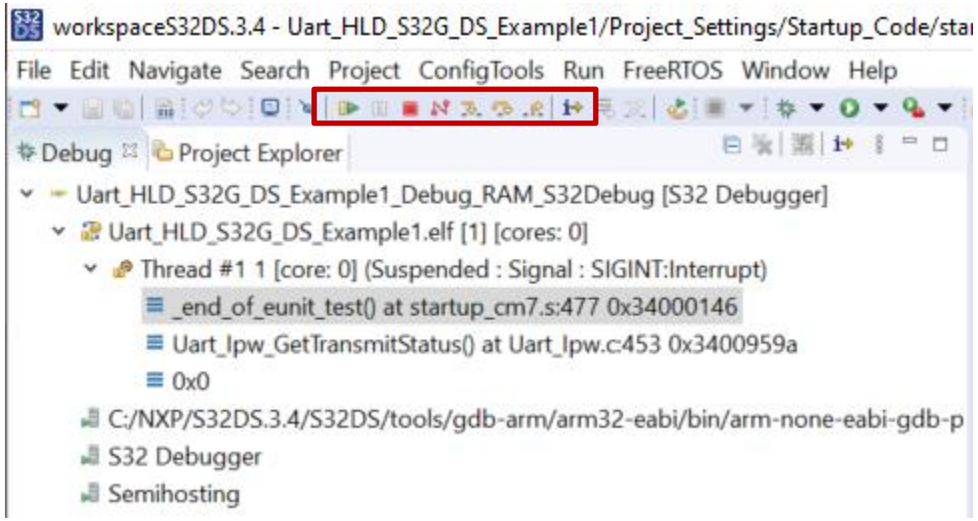
Server port number: 4500

□ Enable log

GDB Client

Executable: ${S32DS_GDB_ARM32_PY}

Commands:

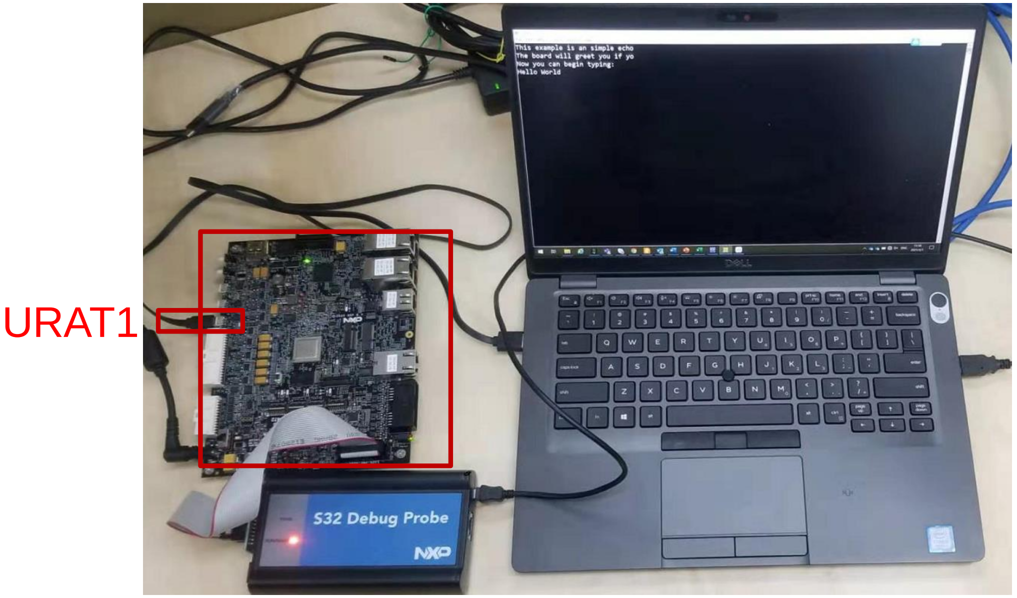


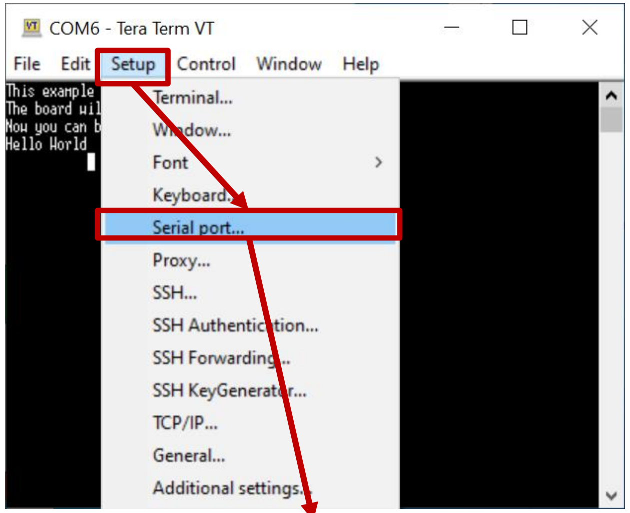
workspaceS32DS.3.4 - Uart_HLD_S32G_DS_Example1/Project_Settings/Startup_Code/star

```
Uart_HLD_S32G_DS_Example1_Debug_RAM_S32Debug [S32 Debugger]
  Uart_HLD_S32G_DS_Example1.elf [1] [cores: 0]
    Thread #1 1 [core: 0] (Suspended : Signal : SIGINT:Interrupt)
      _end_of_eunit_test() at startup_cm7.s:477 0x34000146
      Uart_Ipw_GetTransmitStatus() at Uart_Ipw.c:453 0x3400959a
      0x0
  C:/NXP/S32DS.3.4/S32DS/tools/gdb-arm/arm32-eabi/bin/arm-none-eabi-gdb-p
  S32 Debugger
  Semihosting
```

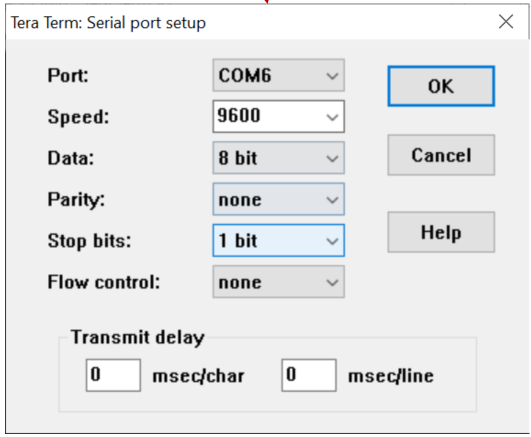
Hands on UART: Test result

- Connect the PC and UART1
- Open Tera Term and Set the serial port
- the terminal software will show the below messages. input “Hello”, UART output “Hello World”





```
This example
The board wil
Now you can b
Hello World
```



Port: COM6

Speed: 9600

Data: 8 bit

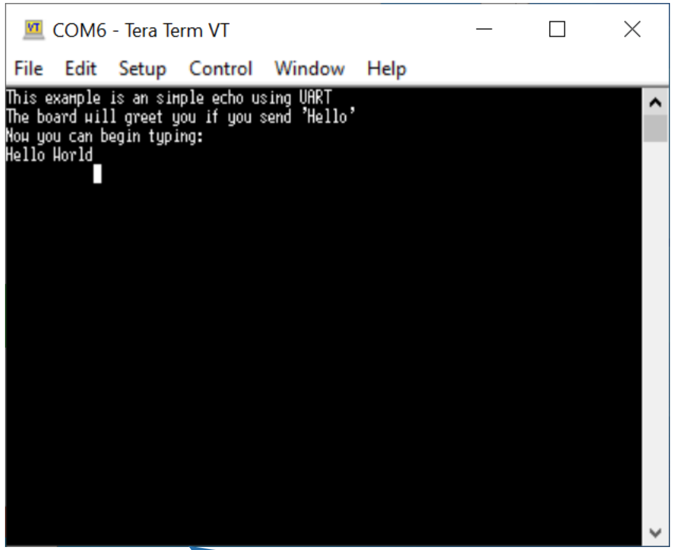
Parity: none

Stop bits: 1 bit

Flow control: none

Transmit delay

0 msec/char 0 msec/line



```
This example is an simple echo using UART
The board will greet you if you send 'Hello'
Now you can begin typing:
Hello World
```

04.

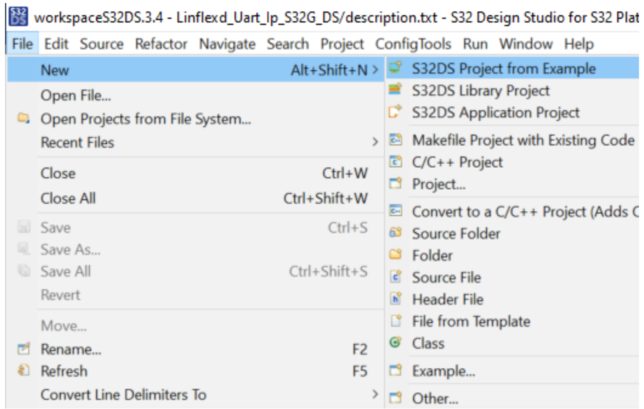
Hands on ETH Example

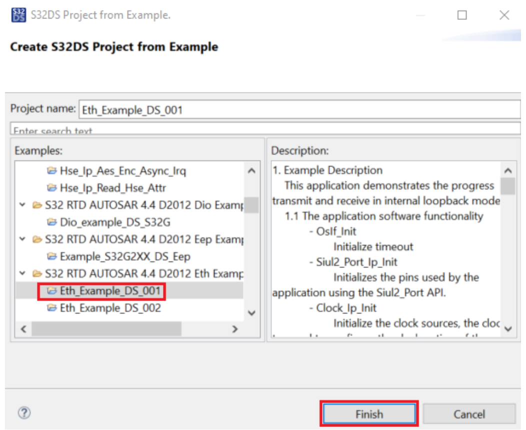
Hands on ETH – Objective

- How to import the ETH example into S32DS
- How to configure the clock of ETH via S32DS
- How to configure the port of ETH via S32DS
- How to use the ETH module to transmit/receive ETH frame
- How to debug the ETH example with S32 debug probe

Hands on ETH: Import ETH example project

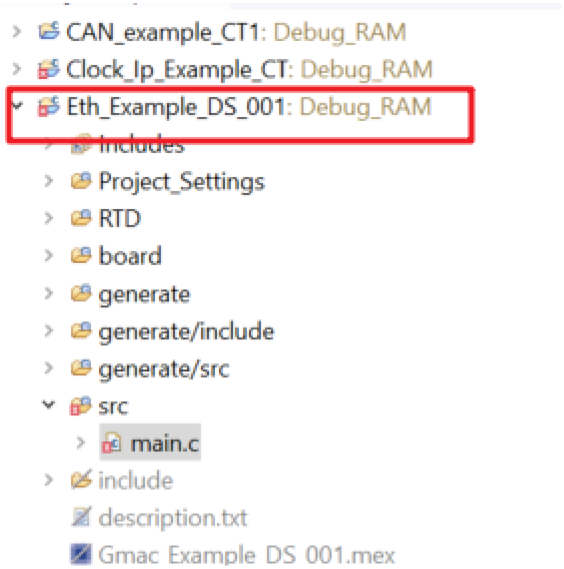
Open S32 Design Studio, go to “File -> New -> S32DS Project From Example”. Select “**Eth_Example_DS_001**” example, then click on “Finish”. The project is copied into current workspace.


workspaceS32DS.3.4 - Linflexd_Uart_Ip_S32G_DS/description.txt - S32 Design Studio for S32 Plat
File Edit Source Refactor Navigate Search Project ConfigTools Run Window Help
New Alt+Shift+N >
Open File...
Open Projects from File System...
Recent Files >
Close Ctrl+W
Close All Ctrl+Shift+W
Save Ctrl+S
Save As...
Save All Ctrl+Shift+S
Revert
Move...
Rename... F2
Refresh F5
Convert Line Delimiters To >
S32DS Project from Example
S32DS Library Project
S32DS Application Project
Makefile Project with Existing Code
C/C++ Project
Project...
Convert to a C/C++ Project (Adds C
Source Folder
Folder
Source File
Header File
File from Template
Class
Example...
Other...

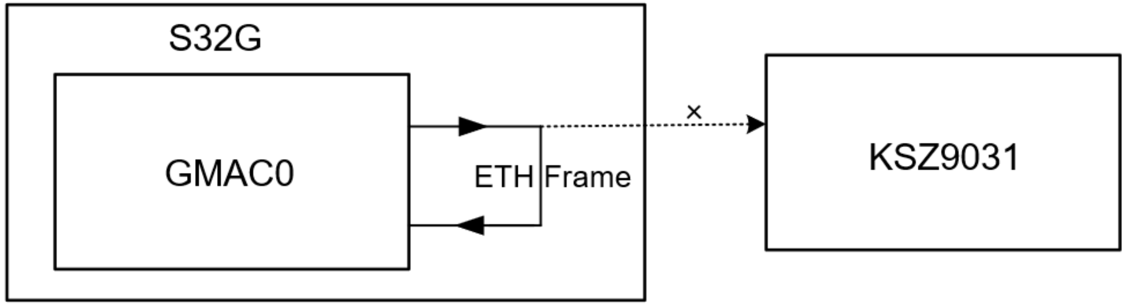


Project name: Eth_Example_DS_001

1. Example Description
   This application demonstrates the progress transmit and receive in internal loopback mode
   1.1 The application software functionality
      - OsIf_Init
         Initialize timeout
      - Siul2_Port_Ip_Init
         Initializes the pins used by the application using the Siul2_Port API.
      - Clock_Ip_Init
         Initialize the clock sources, the cloc



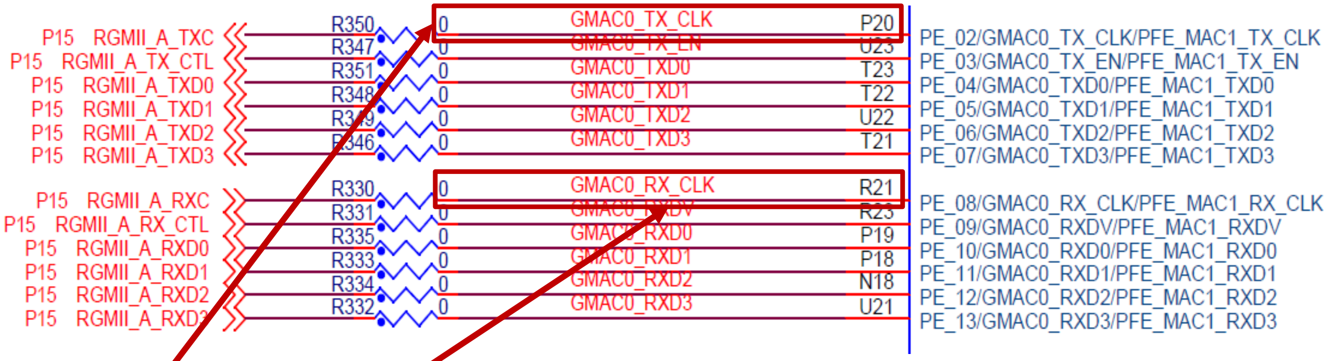
This “Gmac_Example_DS_001” example demonstrates the GMAC transmit and receive in internal loopback mode. The ETH frame is transmitted back directly through GMAC, and the frame will not be transmitted to PHY.

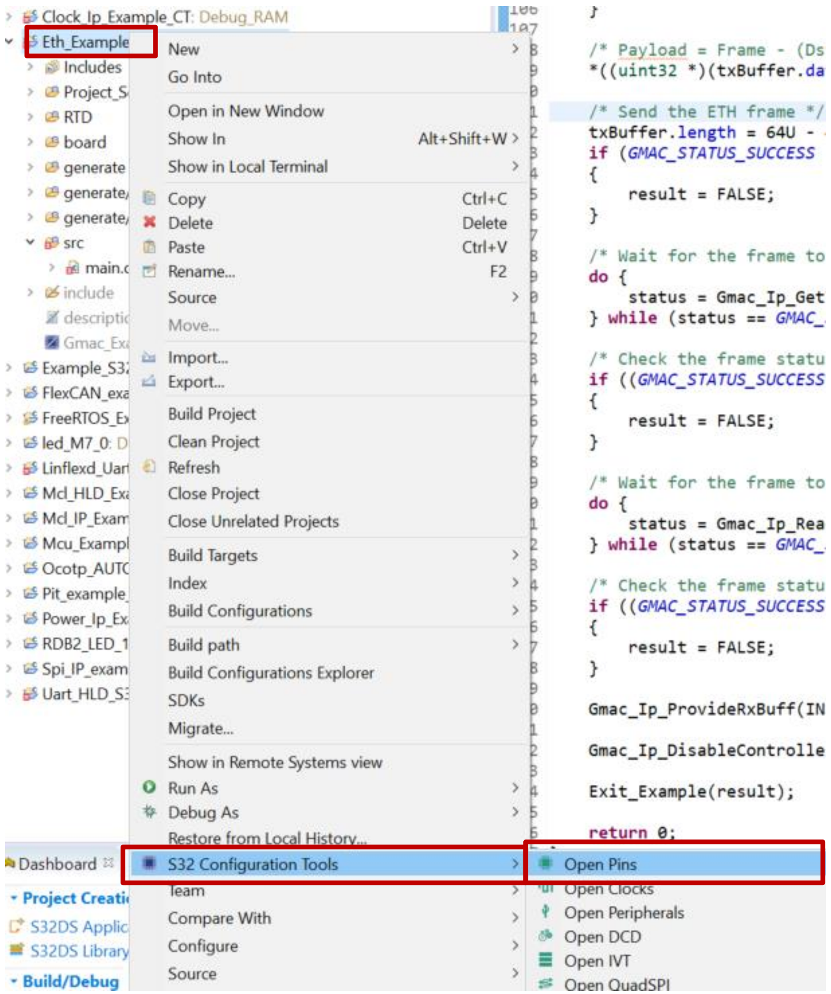


Hands on ETH : Port Configuration 1

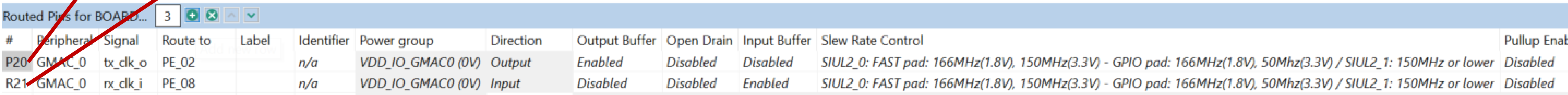
Go to desired configuration tool:

- Right click on Project,
- Select S32 Configuration Tool…
- Select Open Pins
- Configure pins to provide the external clock to Tx, Rx signals





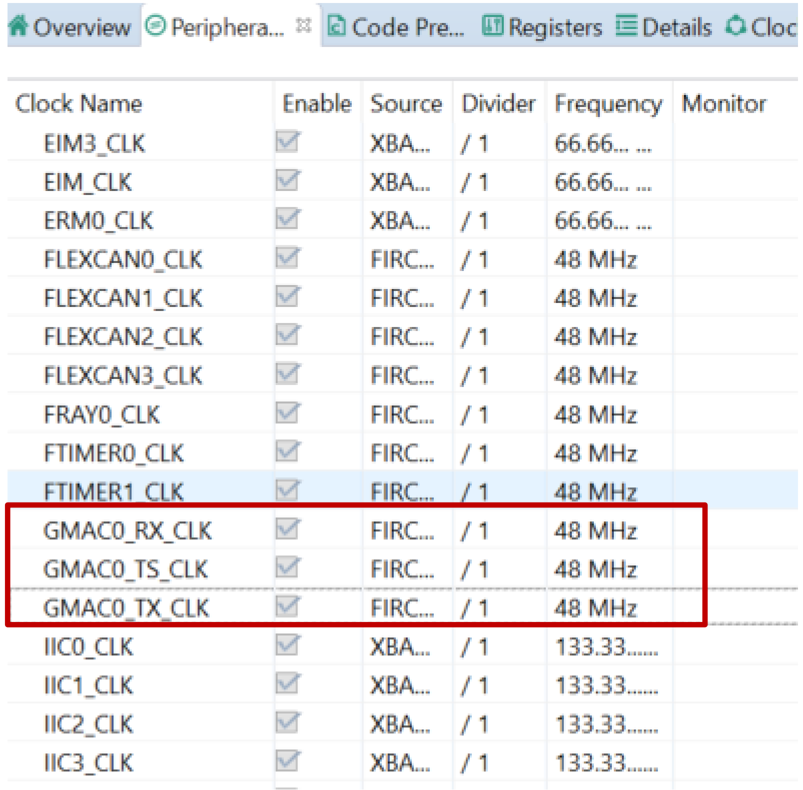
```
    }

    /* Payload = Frame - (Ds
    *((uint32 *)(txBuffer.da

    /* Send the ETH frame */
    txBuffer.length = 64U -
    if (GMAC_STATUS_SUCCESS
    {
        result = FALSE;
    }

    /* Wait for the frame to
    do {
        status = Gmac_Ip_Get
    } while (status == GMAC_

    /* Check the frame statu
    if ((GMAC_STATUS_SUCCESS
    {
        result = FALSE;
    }

    /* Wait for the frame to
    do {
        status = Gmac_Ip_Rea
    } while (status == GMAC_

    /* Check the frame statu
    if ((GMAC_STATUS_SUCCESS
    {
        result = FALSE;
    }

    Gmac_Ip_ProvideRxBuff(IN

    Gmac_Ip_DisableControlle

    Exit_Example(result);

    return 0;
```


Routed Pins for BOARD... 3

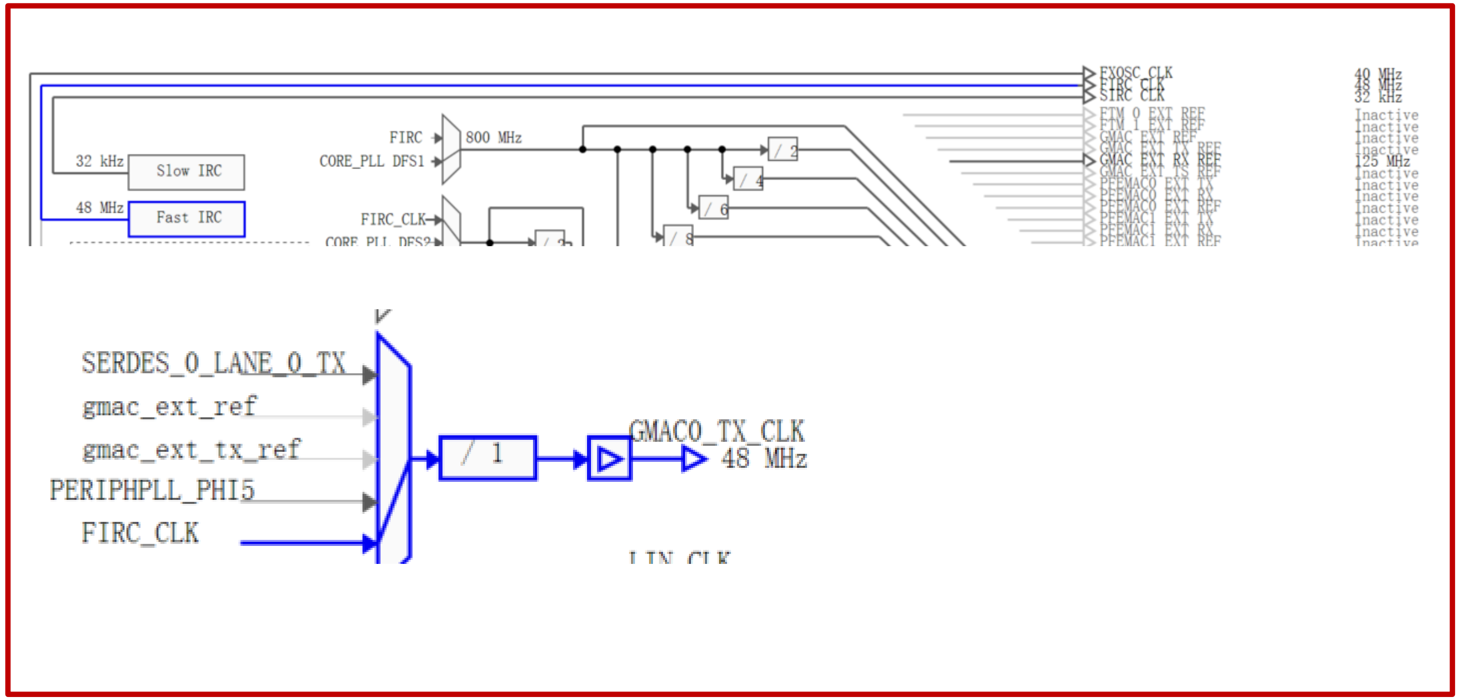
#	Peripheral	Signal	Route to	Label	Identifier	Power group	Direction	Output Buffer	Open Drain	Input Buffer	Slew Rate Control	Pullup Enab
P20	GMAC_0	tx_clk_o	PE_02		n/a	VDD_IO_GMAC0 (0V)	Output	Enabled	Disabled	Disabled	SIUL2_0: FAST pad: 166MHz(1.8V), 150MHz(3.3V) - GPIO pad: 166MHz(1.8V), 50Mhz(3.3V) / SIUL2_1: 150MHz or lower	Disabled
R21	GMAC_0	rx_clk_i	PE_08		n/a	VDD_IO_GMAC0 (0V)	Input	Disabled	Disabled	Enabled	SIUL2_0: FAST pad: 166MHz(1.8V), 150MHz(3.3V) - GPIO pad: 166MHz(1.8V), 50Mhz(3.3V) / SIUL2_1: 150MHz or lower	Disabled

Hands on ETH : Clock Configuration 1

Open the **Peripheral Clock View**, Double click the GMAC0 module. The **Clocks Diagram** shows the power tree of GMAC module



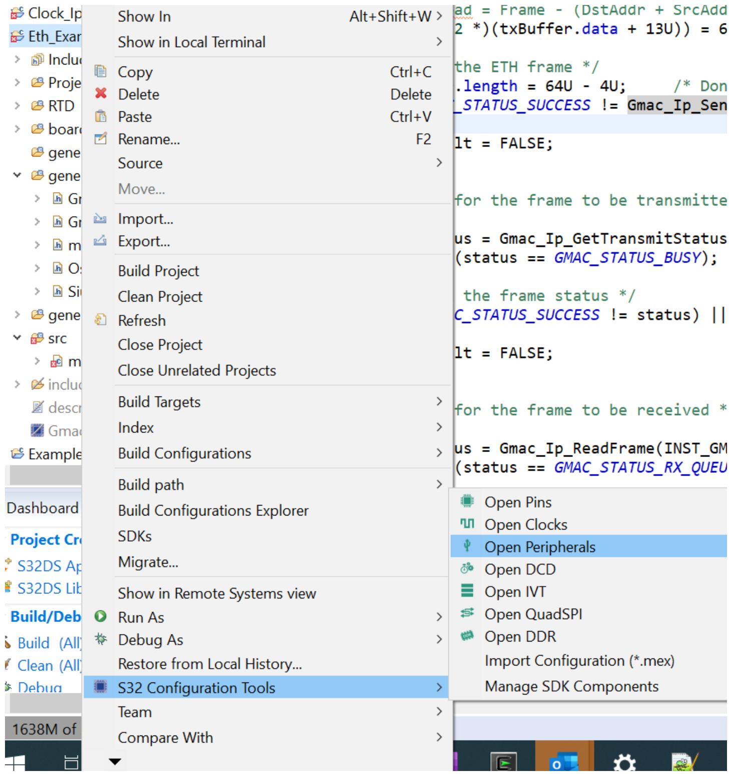
Clock Name	Enable	Source	Divider	Frequency	Monitor
EIM3_CLK	☑	XBA...	/ 1	66.66... ...	
EIM_CLK	☑	XBA...	/ 1	66.66... ...	
ERM0_CLK	☑	XBA...	/ 1	66.66... ...	
FLEXCAN0_CLK	☑	FIRC...	/ 1	48 MHz	
FLEXCAN1_CLK	☑	FIRC...	/ 1	48 MHz	
FLEXCAN2_CLK	☑	FIRC...	/ 1	48 MHz	
FLEXCAN3_CLK	☑	FIRC...	/ 1	48 MHz	
FRAY0_CLK	☑	FIRC...	/ 1	48 MHz	
FTIMER0_CLK	☑	FIRC...	/ 1	48 MHz	
FTIMER1_CLK	☑	FIRC...	/ 1	48 MHz	
GMAC0_RX_CLK	☑	FIRC...	/ 1	48 MHz	
GMAC0_TS_CLK	☑	FIRC...	/ 1	48 MHz	
GMAC0_TX_CLK	☑	FIRC...	/ 1	48 MHz	
IIC0_CLK	☑	XBA...	/ 1	133.33......	
IIC1_CLK	☑	XBA...	/ 1	133.33......	
IIC2_CLK	☑	XBA...	/ 1	133.33......	
IIC3_CLK	☑	XBA...	/ 1	133.33......	



Hands on ETH: ETH configuration

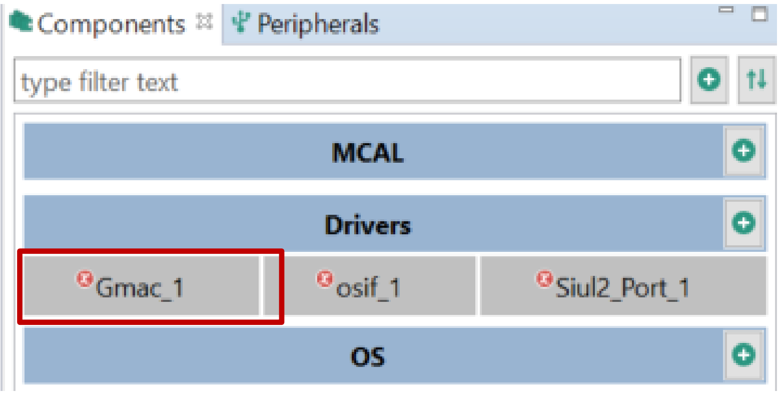
Open the peripheral configuration:

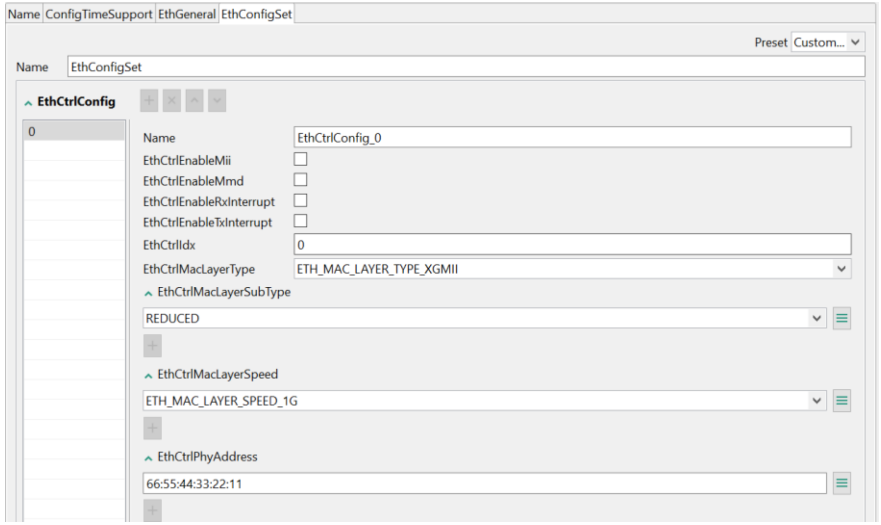
- Right click on Project,
- Select S32 Configuration Tool…
- Select Peripherals



```
ad = Frame - (DstAddr + SrcAdd
2 *)(txBuffer.data + 13U)) = 6

the ETH frame */
.length = 64U - 4U;      /* Don
_STATUS_SUCCESS != Gmac_Ip_Sen

lt = FALSE;

for the frame to be transmitte

us = Gmac_Ip_GetTransmitStatus
(status == GMAC_STATUS_BUSY);

 the frame status */
C_STATUS_SUCCESS != status) ||

lt = FALSE;

for the frame to be received *

us = Gmac_Ip_ReadFrame(INST_GM
(status == GMAC_STATUS_RX_QUEU
```

Select **Components** to find out **GMAC_1** Driver and double click





Preset Custom...

Name EthConfigSet

0

Name EthCtrlConfig_0

EthCtrlEnableMii □

EthCtrlEnableMmd □

EthCtrlEnableRxInterrupt □

EthCtrlEnableTxInterrupt □

EthCtrlIdx 0

EthCtrlMacLayerType ETH_MAC_LAYER_TYPE_XGMII

EthCtrlMacLayerSubType

REDUCED

EthCtrlMacLayerSpeed

ETH_MAC_LAYER_SPEED_1G

EthCtrlPhyAddress

66:55:44:33:22:11

Hands on ETH: Update code

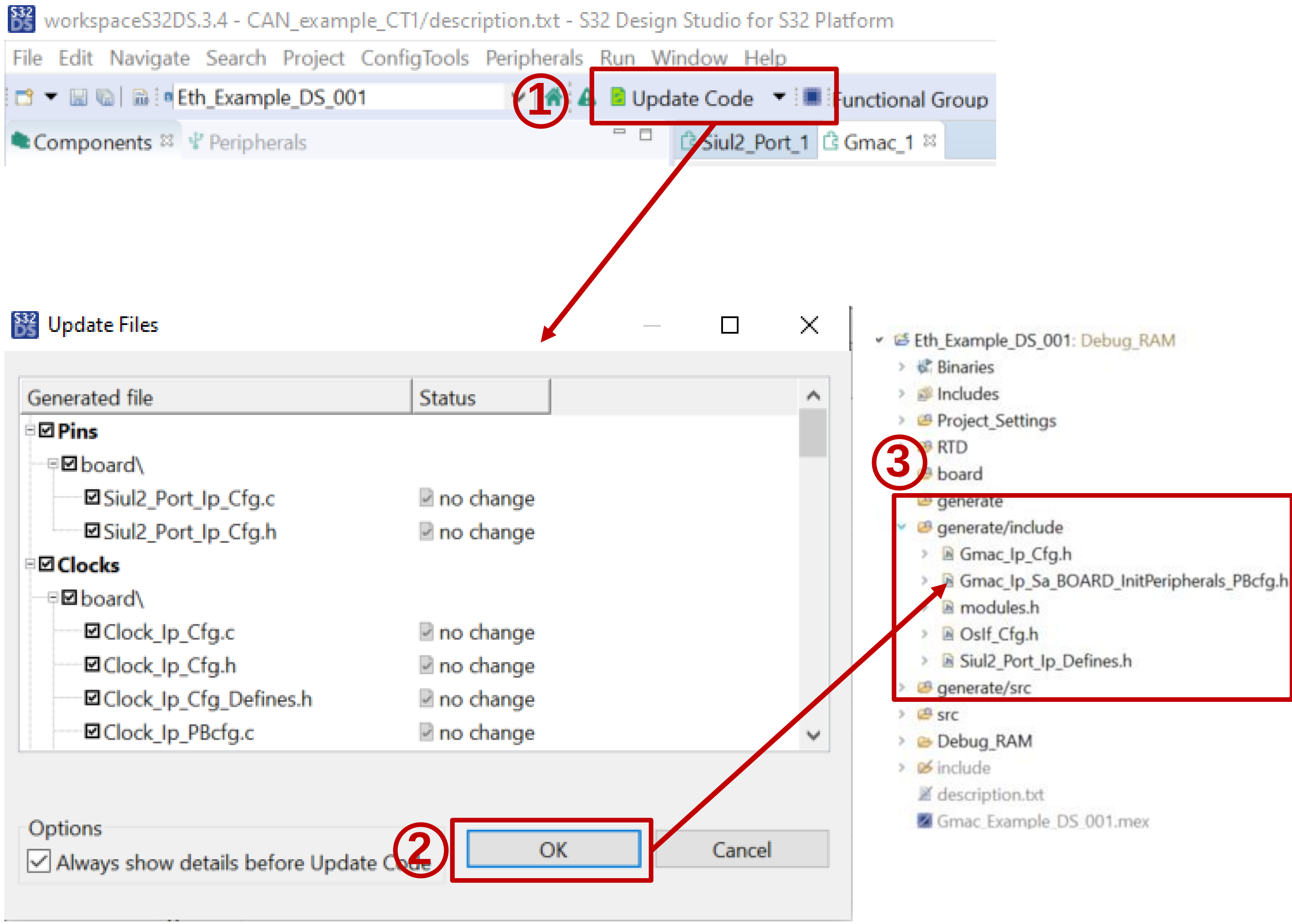
Generate code method:

1.Click on any configuration tool, like Pins

Then click **Update Code** (ensure desired project is selected!)

2.The Update Files widow pops up. It shows the detail update information. Click **ok** button.

3.The configuration .c and .h file will be generated at “generate” folder.



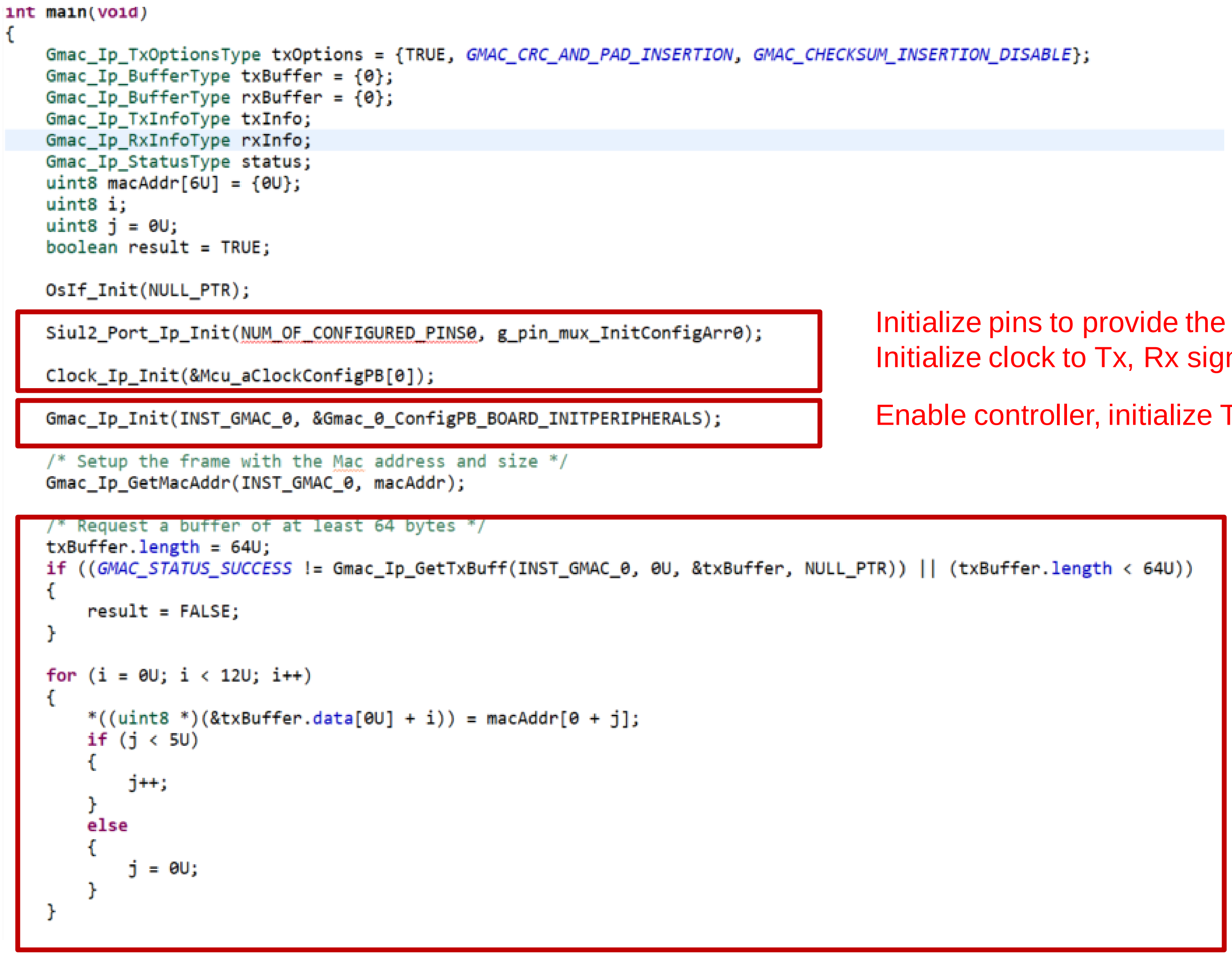
workspaceS32DS.3.4 - CAN_example_CT1/description.txt - S32 Design Studio for S32 Platform

Generated file	Status
☑ **Pins**	
☑ board\	
☑ Siul2_Port_Ip_Cfg.c	no change
☑ Siul2_Port_Ip_Cfg.h	no change
☑ **Clocks**	
☑ board\	
☑ Clock_Ip_Cfg.c	no change
☑ Clock_Ip_Cfg.h	no change
☑ Clock_Ip_Cfg_Defines.h	no change
☑ Clock_Ip_PBcfg.c	no change

Options

☑ Always show details before Update Code

Hands on ETH: Application code 1



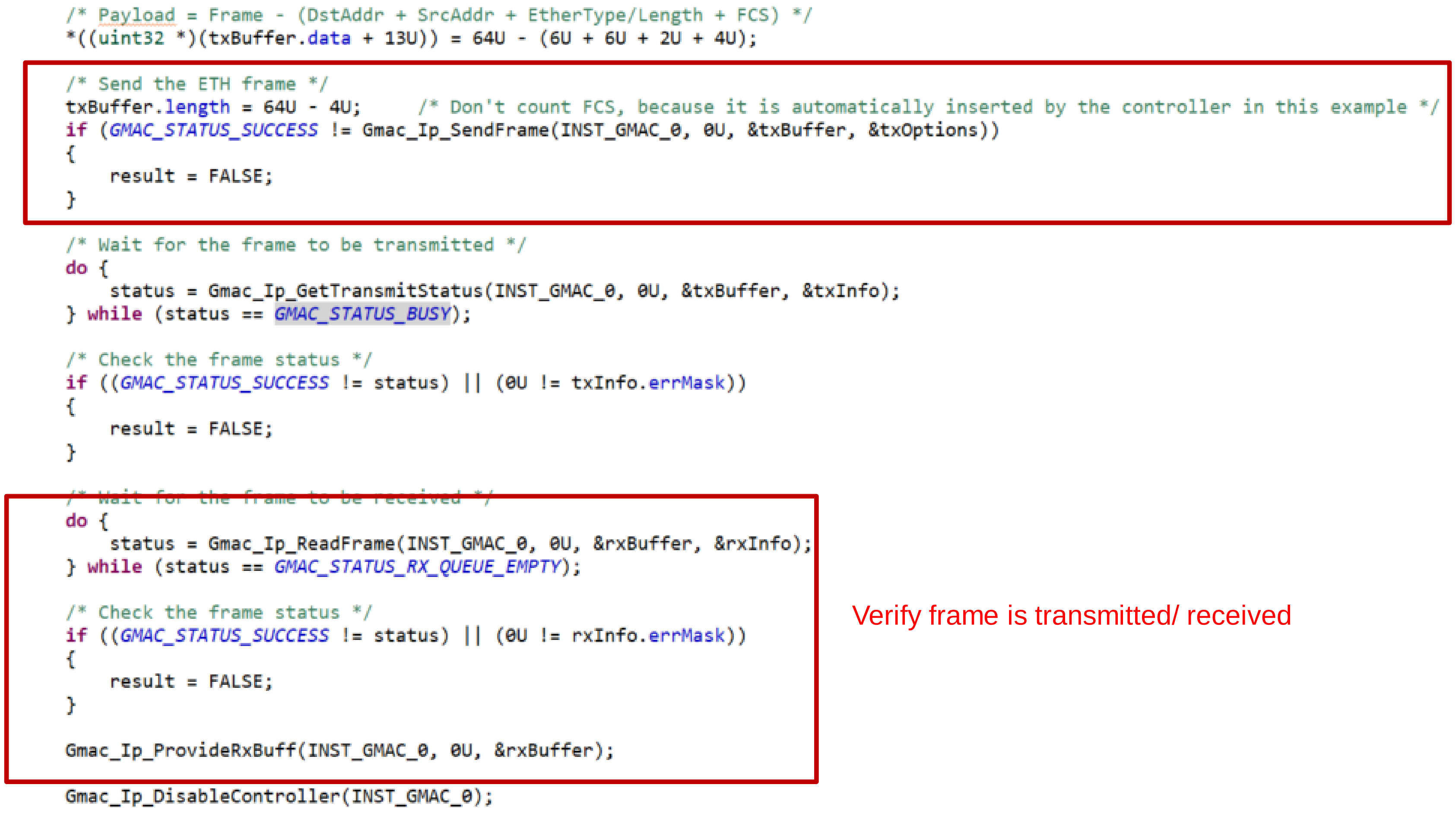
```
int main(void)
{
    Gmac_Ip_TxOptionsType txOptions = {TRUE, GMAC_CRC_AND_PAD_INSERTION, GMAC_CHECKSUM_INSERTION_DISABLE};
    Gmac_Ip_BufferType txBuffer = {0};
    Gmac_Ip_BufferType rxBuffer = {0};
    Gmac_Ip_TxInfoType txInfo;
    Gmac_Ip_RxInfoType rxInfo;
    Gmac_Ip_StatusType status;
    uint8 macAddr[6U] = {0U};
    uint8 i;
    uint8 j = 0U;
    boolean result = TRUE;

    OsIf_Init(NULL_PTR);

    Siul2_Port_Ip_Init(NUM_OF_CONFIGURED_PINS0, g_pin_mux_InitConfigArr0);

    Clock_Ip_Init(&Mcu_aClockConfigPB[0]);

    Gmac_Ip_Init(INST_GMAC_0, &Gmac_0_ConfigPB_BOARD_INITPERIPHERALS);

    /* Setup the frame with the Mac address and size */
    Gmac_Ip_GetMacAddr(INST_GMAC_0, macAddr);

    /* Request a buffer of at least 64 bytes */
    txBuffer.length = 64U;
    if ((GMAC_STATUS_SUCCESS != Gmac_Ip_GetTxBuff(INST_GMAC_0, 0U, &txBuffer, NULL_PTR)) || (txBuffer.length < 64U))
    {
        result = FALSE;
    }

    for (i = 0U; i < 12U; i++)
    {
        *((uint8 *)(&txBuffer.data[0U] + i)) = macAddr[0 + j];
        if (j < 5U)
        {
            j++;
        }
        else
        {
            j = 0U;
        }
    }
```

Initialize pins to provide the external clock to Tx, Rx signals via the function Siul2_Port_Ip_Init
Initialize clock to Tx, Rx signals via the function Clock_Ip_Init

Enable controller, initialize Tx and Rx buffer via the function Gmac_Ip_Init

initialize transmit buffer and Borrow transmit area to load frame via the function Gmac_Ip_GetTxBuff

Hands on ETH: Application code 2



```
/* Payload = Frame - (DstAddr + SrcAddr + EtherType/Length + FCS) */
*((uint32 *)(txBuffer.data + 13U)) = 64U - (6U + 6U + 2U + 4U);

/* Send the ETH frame */
txBuffer.length = 64U - 4U;     /* Don't count FCS, because it is automatically inserted by the controller in this example */
if (GMAC_STATUS_SUCCESS != Gmac_Ip_SendFrame(INST_GMAC_0, 0U, &txBuffer, &txOptions))
{
    result = FALSE;
}

/* Wait for the frame to be transmitted */
do {
    status = Gmac_Ip_GetTransmitStatus(INST_GMAC_0, 0U, &txBuffer, &txInfo);
} while (status == GMAC_STATUS_BUSY);

/* Check the frame status */
if ((GMAC_STATUS_SUCCESS != status) || (0U != txInfo.errMask))
{
    result = FALSE;
}

/* Wait for the frame to be received */
do {
    status = Gmac_Ip_ReadFrame(INST_GMAC_0, 0U, &rxBuffer, &rxInfo);
} while (status == GMAC_STATUS_RX_QUEUE_EMPTY);

/* Check the frame status */
if ((GMAC_STATUS_SUCCESS != status) || (0U != rxInfo.errMask))
{
    result = FALSE;
}

Gmac_Ip_ProvideRxBuff(INST_GMAC_0, 0U, &rxBuffer);

Gmac_Ip_DisableController(INST_GMAC_0);
```

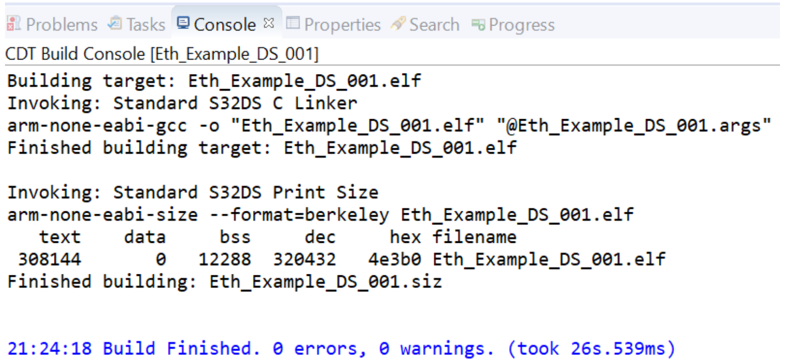
Trigger the transmit frame via Gmac_Ip_SendFrame

Verify frame is transmitted/ received

Hands on ETH: Build and Debug 1

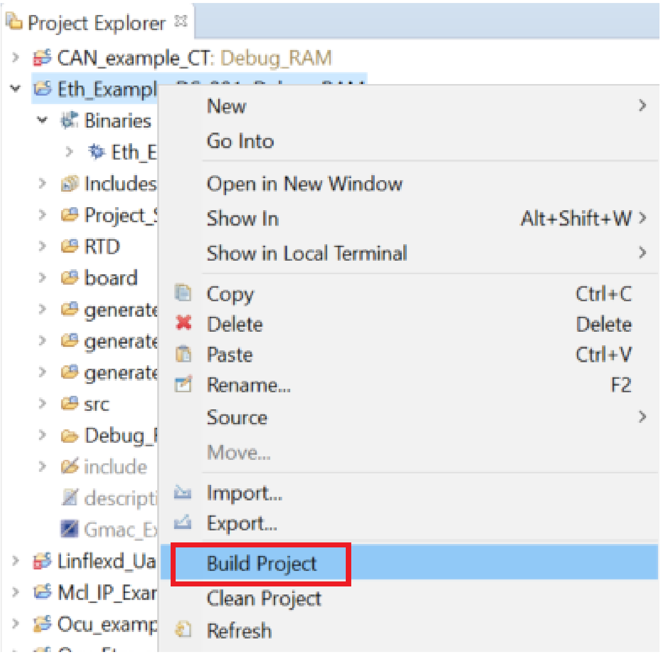
Build target Project:

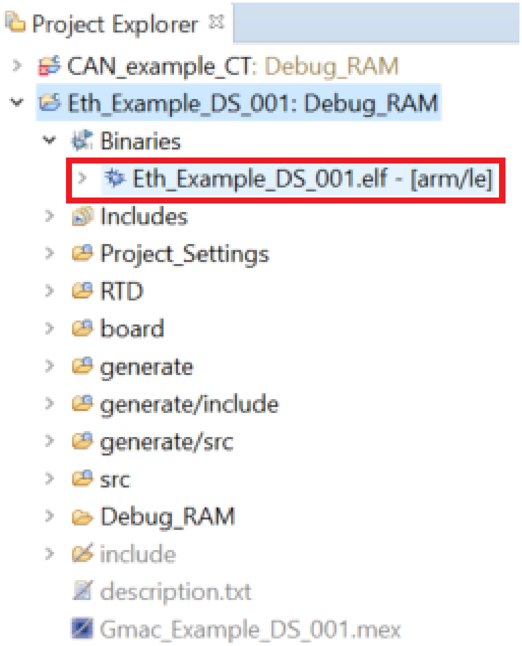
- Right click on Project,
- Build Project
- The console print build information
- Eth_Example_DS_001.elf is created



CDT Build Console [Eth_Example_DS_001]

```
Building target: Eth_Example_DS_001.elf
Invoking: Standard S32DS C Linker
arm-none-eabi-gcc -o "Eth_Example_DS_001.elf" "@Eth_Example_DS_001.args"
Finished building target: Eth_Example_DS_001.elf

Invoking: Standard S32DS Print Size
arm-none-eabi-size --format=berkeley Eth_Example_DS_001.elf
   text    data     bss     dec     hex filename
 308144       0   12288  320432   4e3b0 Eth_Example_DS_001.elf
Finished building: Eth_Example_DS_001.siz


21:24:18 Build Finished. 0 errors, 0 warnings. (took 26s.539ms)
```

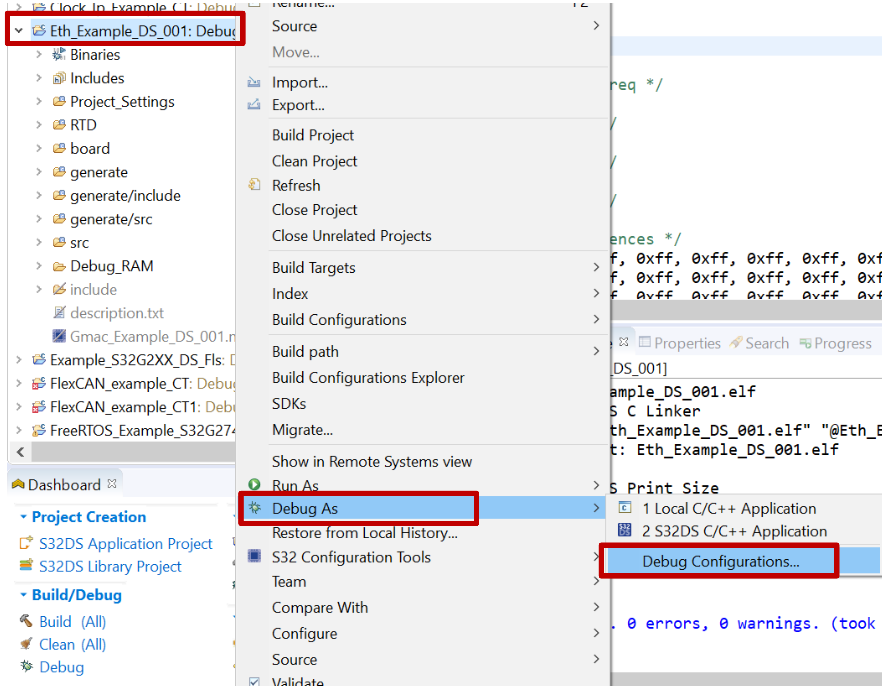




Hands on ETH: Build and Debug 2

Go to debug configuration:

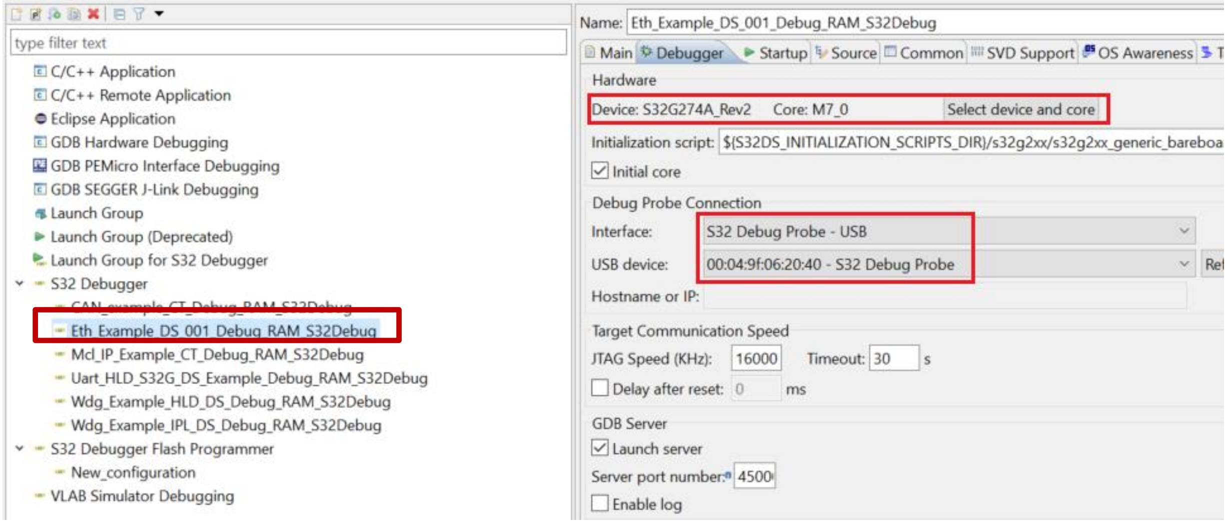
- Right click on Project,
- Select the Debug As
- Click Configurations



```
req */
ences */
f, 0xff, 0xff, 0xff, 0xff, 0xf
f, 0xff, 0xff, 0xff, 0xff, 0xf
DS_001]
ample_DS_001.elf
S C Linker
th_Example_DS_001.elf" "@Eth_E
t: Eth_Example_DS_001.elf
S Print Size
. 0 errors, 0 warnings. (took
```

Debug configuration set:

- Click target project ,
- Select the target device
- Select target S32 Debug Probe


type filter text
C/C++ Application
C/C++ Remote Application
Eclipse Application
GDB Hardware Debugging
GDB PEMicro Interface Debugging
GDB SEGGER J-Link Debugging
Launch Group
Launch Group (Deprecated)
Launch Group for S32 Debugger
S32 Debugger
CAN_example_CT_Debug_RAM_S32Debug
Eth_Example_DS_001_Debug_RAM_S32Debug
Mcl_IP_Example_CT_Debug_RAM_S32Debug
Uart_HLD_S32G_DS_Example_Debug_RAM_S32Debug
Wdg_Example_HLD_DS_Debug_RAM_S32Debug
Wdg_Example_IPL_DS_Debug_RAM_S32Debug
S32 Debugger Flash Programmer
New_configuration
VLAB Simulator Debugging
Main
Debugger
Startup
Source
Common
SVD Support
OS Awareness
T
Select device and core
Ref

Name: Eth_Example_DS_001_Debug_RAM_S32Debug

Hardware

Device: S32G274A_Rev2 Core: M7_0

Initialization script: ${S32DS_INITIALIZATION_SCRIPTS_DIR}/s32g2xx/s32g2xx_generic_bareboa

☑ Initial core

Debug Probe Connection

Interface: S32 Debug Probe - USB

USB device: 00:04:9f:06:20:40 - S32 Debug Probe

Hostname or IP:

Target Communication Speed

JTAG Speed (KHz): 16000 Timeout: 30 s

☐ Delay after reset: 0 ms

GDB Server

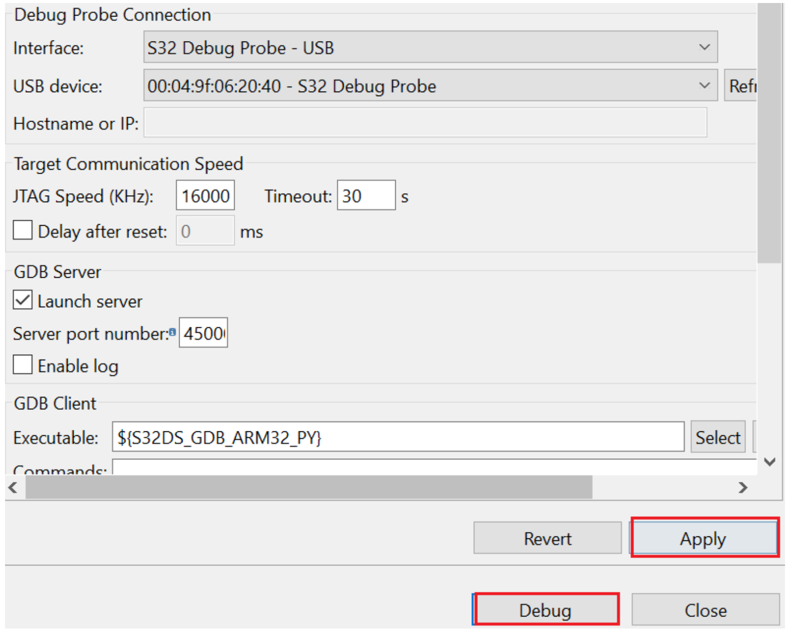
☑ Launch server

Server port number: 4500

☐ Enable log

Hands on ETH: Debug and run

Click on “Apply”, then click on “Debug”. the perspective will jump to the Debug Perspective, and you can use the controls to control the program flow.



Debug Probe Connection

Interface: S32 Debug Probe - USB

USB device: 00:04:9f:06:20:40 - S32 Debug Probe

Hostname or IP:

Target Communication Speed

JTAG Speed (KHz): 16000 Timeout: 30 s

□ Delay after reset: 0 ms

GDB Server

☑ Launch server

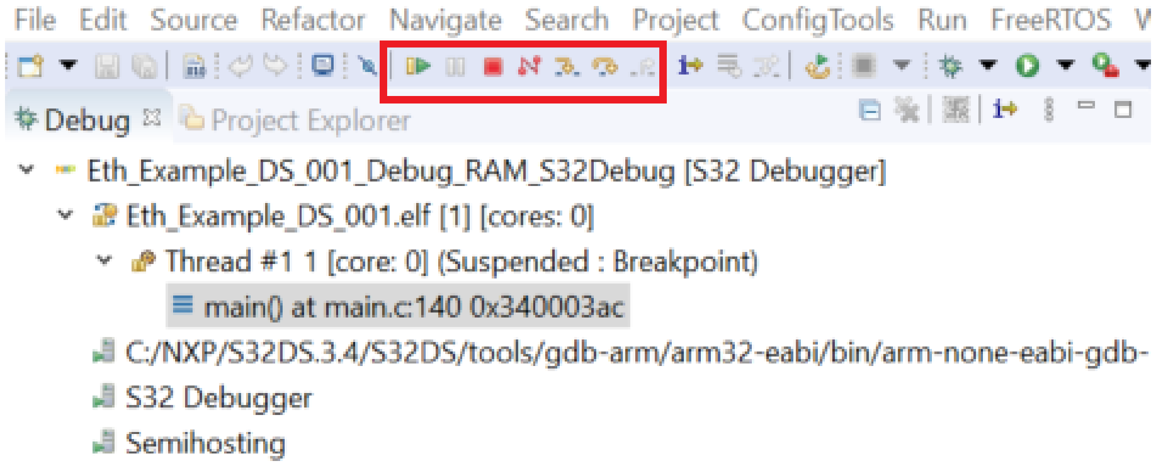
Server port number: 4500

□ Enable log

GDB Client

Executable: ${S32DS_GDB_ARM32_PY}

Commands:

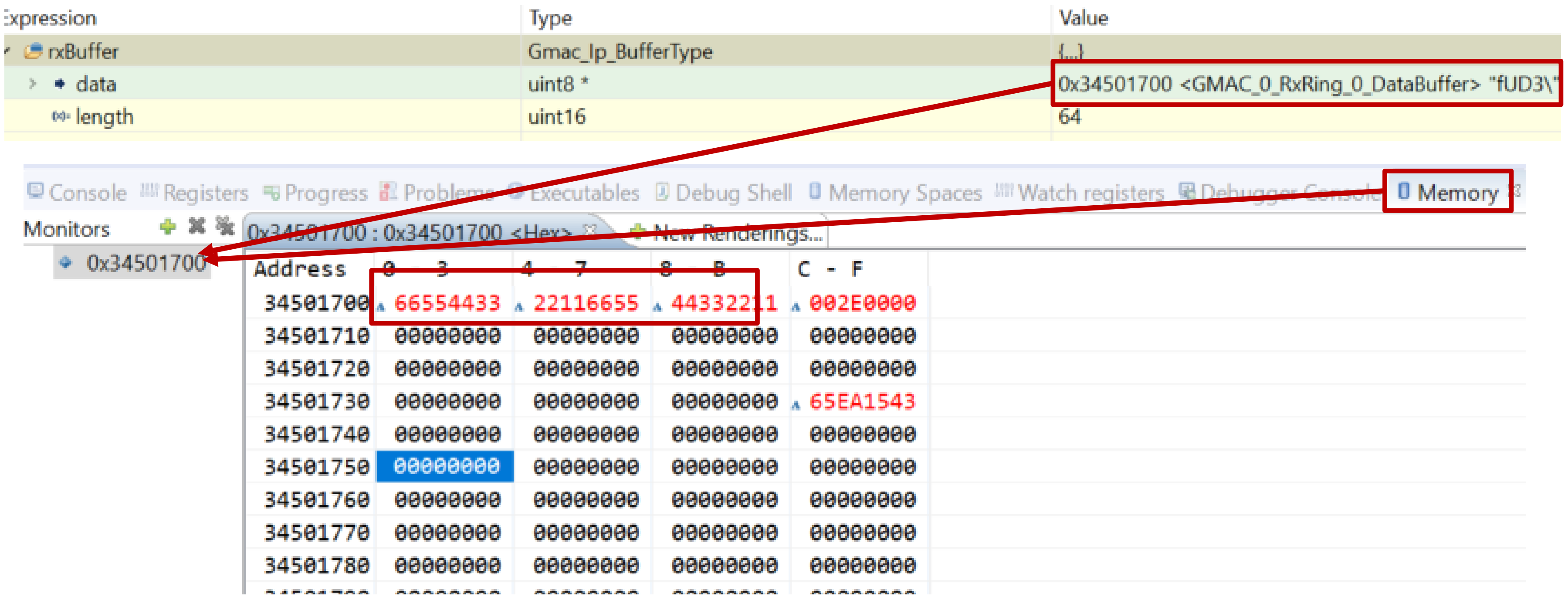


File Edit Source Refactor Navigate Search Project ConfigTools Run FreeRTOS W
Debug
Project Explorer

Eth_Example_DS_001_Debug_RAM_S32Debug [S32 Debugger]
Eth_Example_DS_001.elf [1] [cores: 0]
Thread #1 1 [core: 0] (Suspended : Breakpoint)
main() at main.c:140 0x340003ac
C:/NXP/S32DS.3.4/S32DS/tools/gdb-arm/arm32-eabi/bin/arm-none-eabi-gdb-
S32 Debugger
Semihosting

Hands on ETH: Test result

In this project. The eth frame of Transmit & receive in internal loopback mode. The rxBuffer shows the received frame.



:xpression	Type	Value
rxBuffer	Gmac_Ip_BufferType	{...}
data	uint8 *	0x34501700 <GMAC_0_RxRing_0_DataBuffer> "fUD3\"
length	uint16	64

```
Address   0 - 3     4 - 7     8 - B     C - F
34501700  66554433  22116655  44332211  002E0000
34501710  00000000  00000000  00000000  00000000
34501720  00000000  00000000  00000000  00000000
34501730  00000000  00000000  00000000  65EA1543
34501740  00000000  00000000  00000000  00000000
34501750  00000000  00000000  00000000  00000000
34501760  00000000  00000000  00000000  00000000
34501770  00000000  00000000  00000000  00000000
34501780  00000000  00000000  00000000  00000000
```

03.

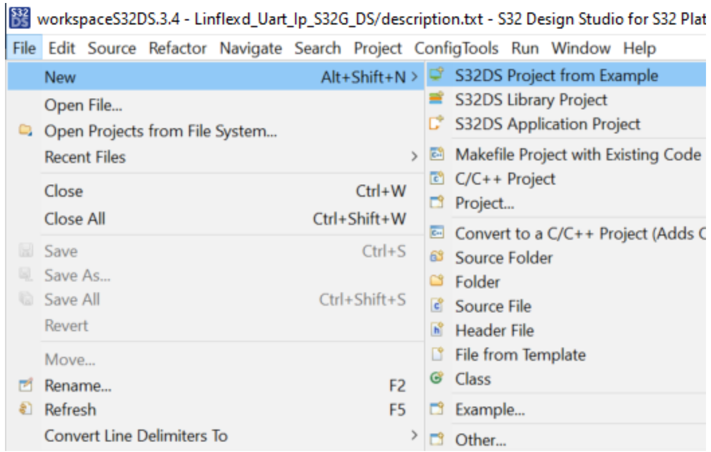
Hands on CAN Example

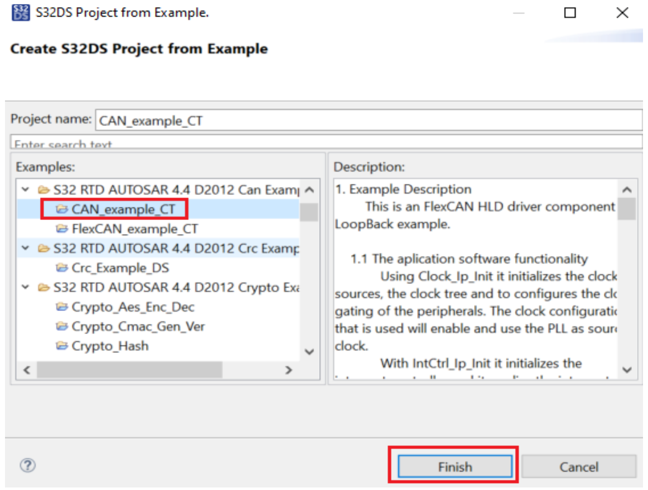
Hands on CAN – Objective

- How to import the CAN example into S32DS
- How to configure the clock of CAN via S32DS
- How to configure the port of CAN via S32DS
- How to modify the CAN loopback
- How to debug the CAN example with S32 debug probe

Hands on CAN : Import CAN example project

Open S32DS3.4, go to “File -> New -> S32DS Project From Example”. Select “**CAN_example_CT**” example, then click on “Finish”. The project is copied into current workspace.


workspaceS32DS.3.4 - Linflexd_Uart_Ip_S32G_DS/description.txt - S32 Design Studio for S32 Plat
File Edit Source Refactor Navigate Search Project ConfigTools Run Window Help
New Alt+Shift+N
Open File...
Open Projects from File System...
Recent Files
Close Ctrl+W
Close All Ctrl+Shift+W
Save Ctrl+S
Save As...
Save All Ctrl+Shift+S
Revert
Move...
Rename... F2
Refresh F5
Convert Line Delimiters To
S32DS Project from Example
S32DS Library Project
S32DS Application Project
Makefile Project with Existing Code
C/C++ Project
Project...
Convert to a C/C++ Project (Adds C
Source Folder
Folder
Source File
Header File
File from Template
Class
Example...
Other...


S32DS Project from Example.
Create S32DS Project from Example
Enter search text
Examples:
S32 RTD AUTOSAR 4.4 D2012 Can Examp
CAN_example_CT
FlexCAN_example_CT
S32 RTD AUTOSAR 4.4 D2012 Crc Examp
Crc_Example_DS
S32 RTD AUTOSAR 4.4 D2012 Crypto Exa
Crypto_Aes_Enc_Dec
Crypto_Cmac_Gen_Ver
Crypto_Hash
Finish
Cancel

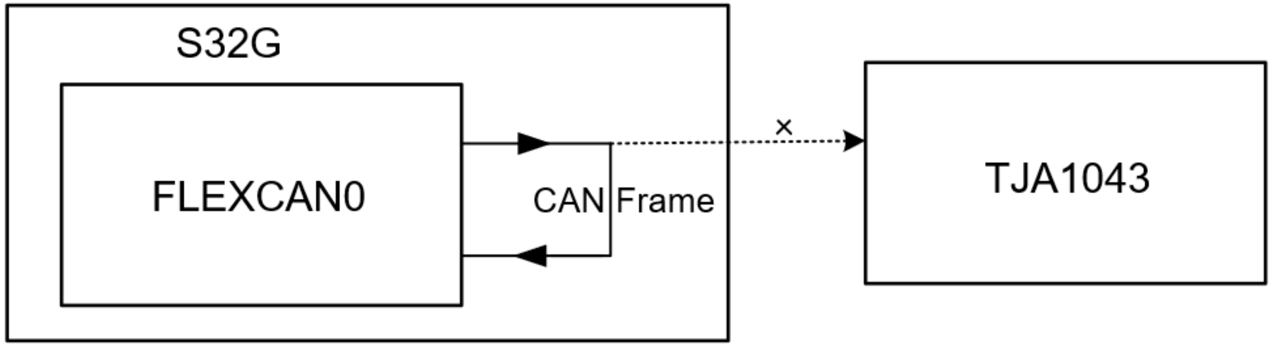
Project name: CAN_example_CT

Description:

1. Example Description
    This is an FlexCAN HLD driver component LoopBack example.

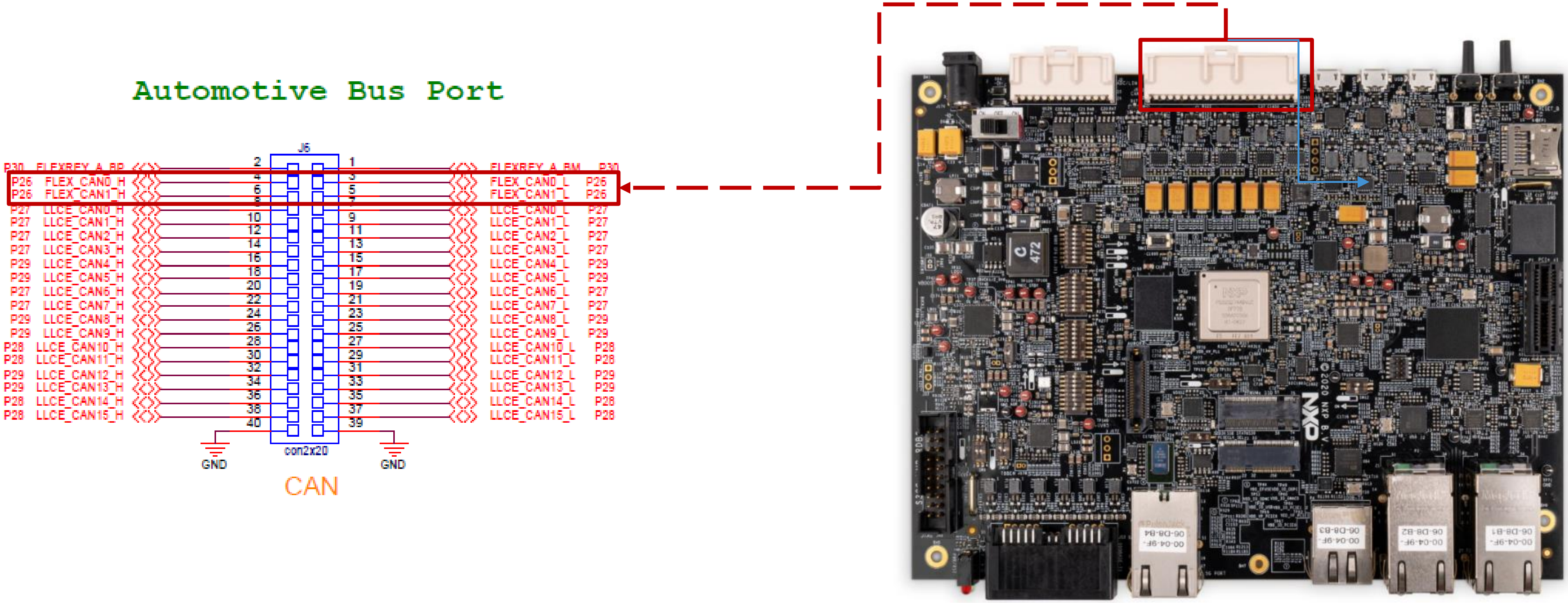
  1.1 The aplication software functionality
      Using Clock_Ip_Init it initializes the clock sources, the clock tree and to configures the cl gating of the peripherals. The clock configurati that is used will enable and use the PLL as sour clock.
      With IntCtrl_Ip_Init it initializes the

“CAN_example_CT” project is a FlexCAN HLD driver component LoopBack project.



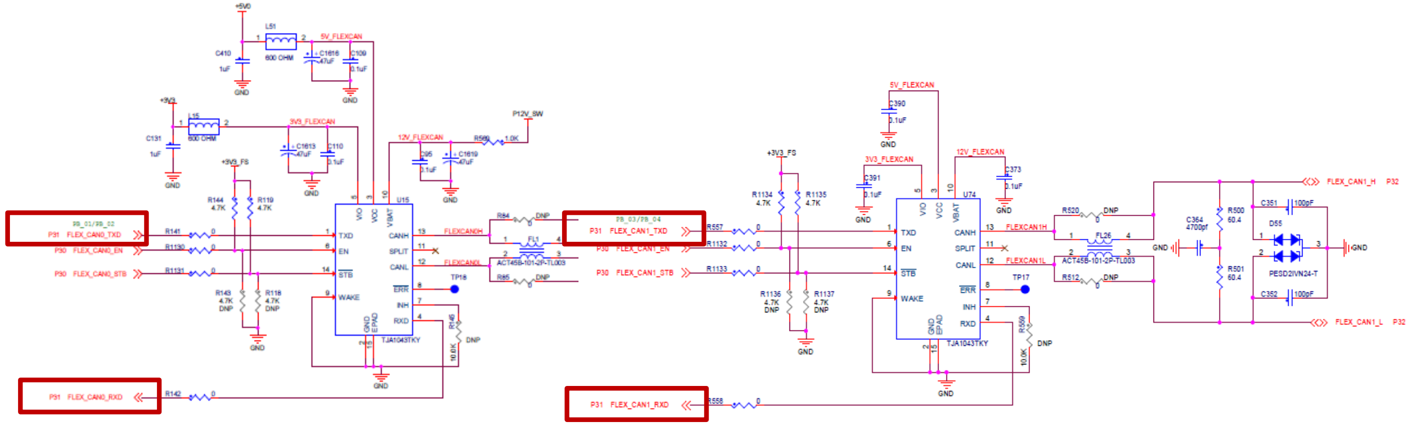
Hands on CAN: the proposed demo need to modify

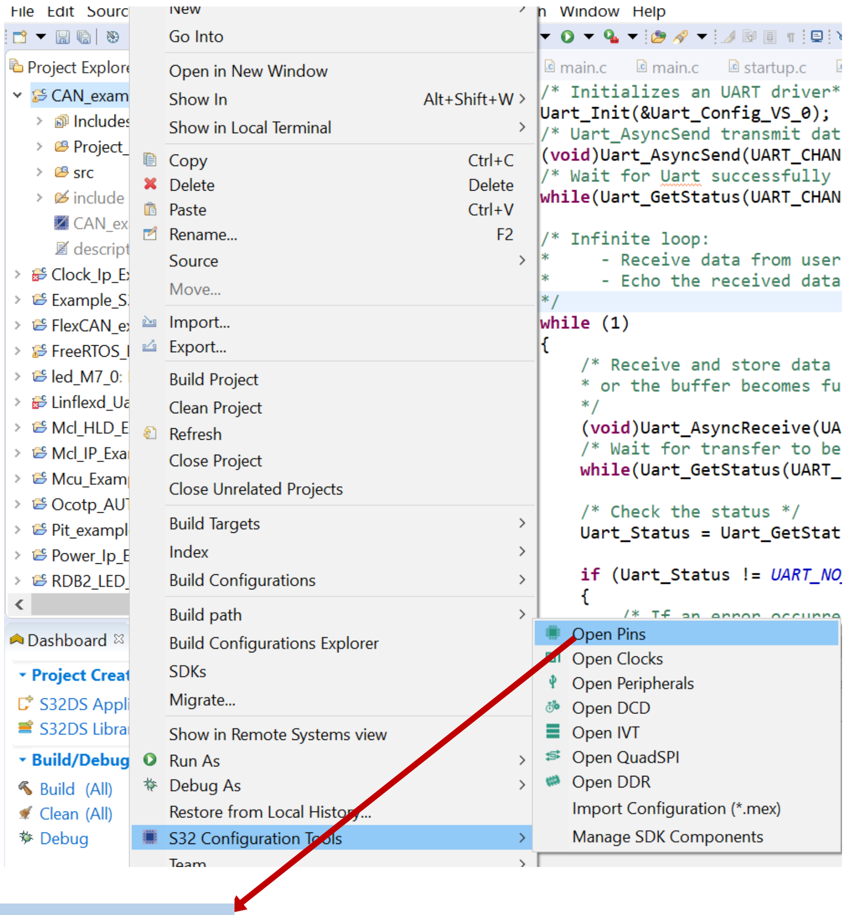
The “CAN_example_CT” project only support loopback model. modify this default project configuration to build transmit/receive CAN frame from FlexCAN_0 to FlexCAN_1


Automotive Bus Port
J6
P30 FLEXRFY_A_BP 2 1 FLEXRFY_A_BM P30
P26 FLEX_CAN0_H 4 3 FLEX_CAN0_L P26
P26 FLEX_CAN1_H 6 5 FLEX_CAN1_L P26
P27 LLCE_CAN0_H 8 7 LLCE_CAN0_L P27
P27 LLCE_CAN1_H 10 9 LLCE_CAN1_L P27
P27 LLCE_CAN2_H 12 11 LLCE_CAN2_L P27
P27 LLCE_CAN3_H 14 13 LLCE_CAN3_L P27
P29 LLCE_CAN4_H 16 15 LLCE_CAN4_L P29
P29 LLCE_CAN5_H 18 17 LLCE_CAN5_L P29
P27 LLCE_CAN6_H 20 19 LLCE_CAN6_L P27
P27 LLCE_CAN7_H 22 21 LLCE_CAN7_L P27
P29 LLCE_CAN8_H 24 23 LLCE_CAN8_L P29
P29 LLCE_CAN9_H 26 25 LLCE_CAN9_L P29
P28 LLCE_CAN10_H 28 27 LLCE_CAN10_L P28
P28 LLCE_CAN11_H 30 29 LLCE_CAN11_L P28
P29 LLCE_CAN12_H 32 31 LLCE_CAN12_L P29
P29 LLCE_CAN13_H 34 33 LLCE_CAN13_L P29
P28 LLCE_CAN14_H 36 35 LLCE_CAN14_L P28
P28 LLCE_CAN15_H 38 37 LLCE_CAN15_L P28
40 39
GND
con2x20
GND
CAN

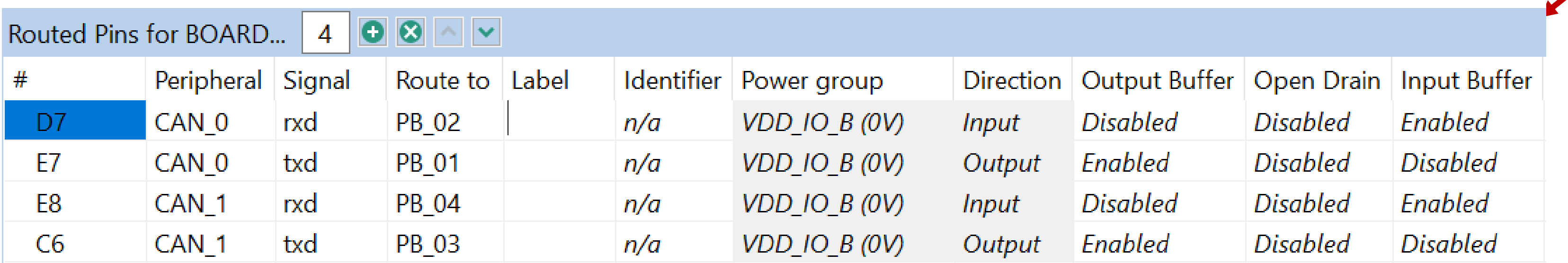
Hands on CAN: Port Configuration 1

- Go to desired configuration tool:
  - Right click on Project,
  - Select S32 Configuration Tool…
  - Select Open Pins
  - Modify the Pins as the schematic of CAN0 and CAN 1





```
/* Initializes an UART driver*
Uart_Init(&Uart_Config_VS_0);
/* Uart_AsyncSend transmit dat
(void)Uart_AsyncSend(UART_CHAN
/* Wait for Uart successfully
while(Uart_GetStatus(UART_CHAN

/* Infinite loop:
*    - Receive data from user
*    - Echo the received data
*/
while (1)
{
    /* Receive and store data
     * or the buffer becomes fu
     */
    (void)Uart_AsyncReceive(UA
    /* Wait for transfer to be
    while(Uart_GetStatus(UART_

    /* Check the status */
    Uart_Status = Uart_GetStat

    if (Uart_Status != UART_NO
    {
        /* If an error occurre
```

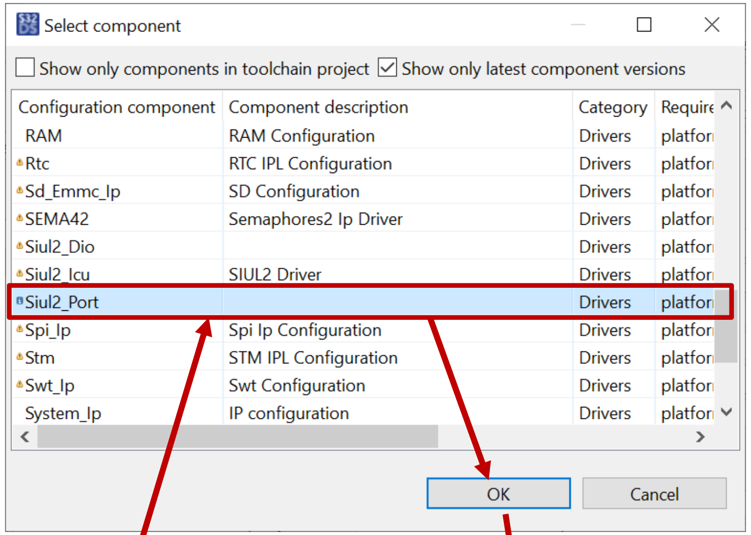


#	Peripheral	Signal	Route to	Label	Identifier	Power group	Direction	Output Buffer	Open Drain	Input Buffer
D7	CAN_0	rxd	PB_02		n/a	VDD_IO_B (0V)	Input	Disabled	Disabled	Enabled
E7	CAN_0	txd	PB_01		n/a	VDD_IO_B (0V)	Output	Enabled	Disabled	Disabled
E8	CAN_1	rxd	PB_04		n/a	VDD_IO_B (0V)	Input	Disabled	Disabled	Enabled
C6	CAN_1	txd	PB_03		n/a	VDD_IO_B (0V)	Output	Enabled	Disabled	Disabled

Hands on CAN: Port Configuration 2

- Add the Port configuration:
  - Right click on Project,
  - Select S32 Configuration Tool…
  - Select Open Peripherals

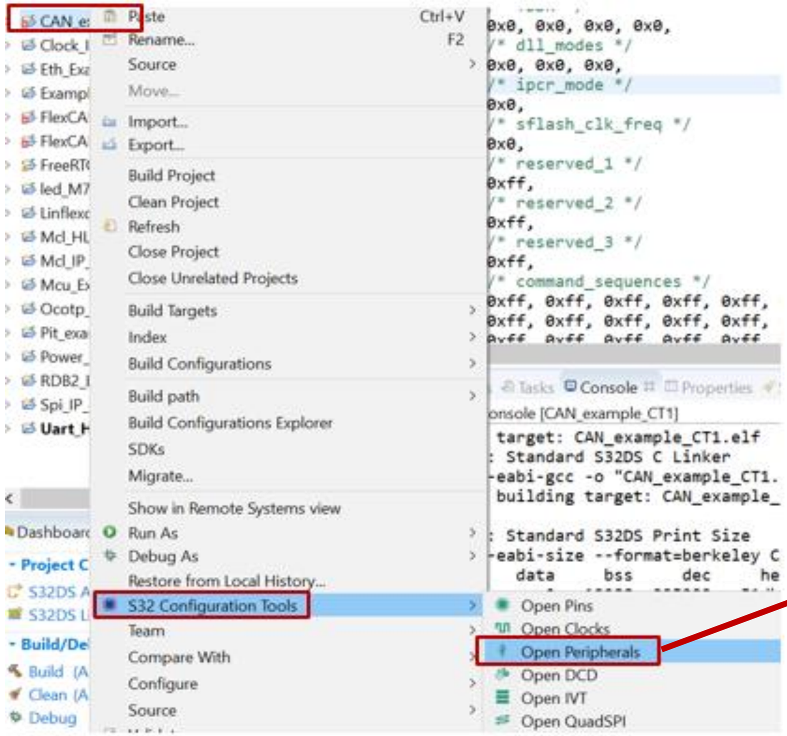
- Click the plus button
- Click the Siul2_Port component
- The Siul2_Port_1 will be added



Select component

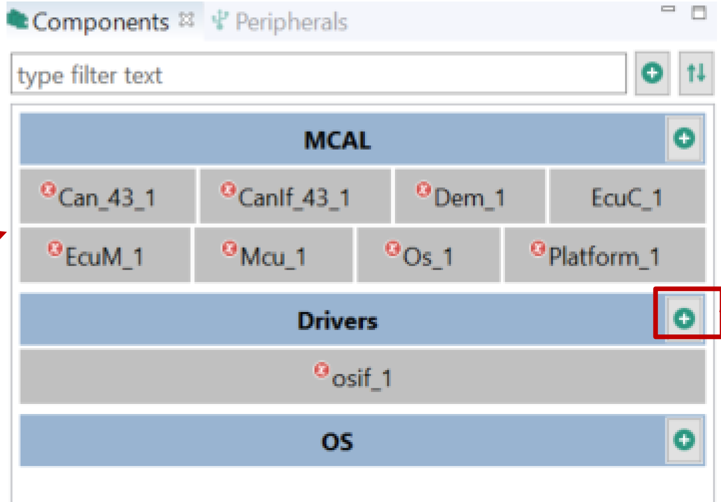
☐ Show only components in toolchain project ☑ Show only latest component versions

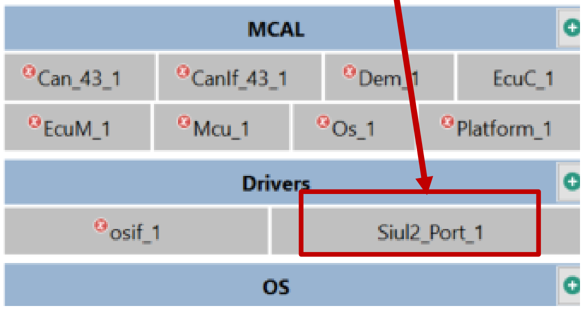
Configuration component	Component description	Category	Require
RAM	RAM Configuration	Drivers	platfor
Rtc	RTC IPL Configuration	Drivers	platfor
Sd_Emmc_Ip	SD Configuration	Drivers	platfor
SEMA42	Semaphores2 Ip Driver	Drivers	platfor
Siul2_Dio		Drivers	platfor
Siul2_Icu	SIUL2 Driver	Drivers	platfor
Siul2_Port		Drivers	platfor
Spi_Ip	Spi Ip Configuration	Drivers	platfor
Stm	STM IPL Configuration	Drivers	platfor
Swt_Ip	Swt Configuration	Drivers	platfor
System_Ip	IP configuration	Drivers	platfor



```
0x0, 0x0, 0x0, 0x0,
/* dll_modes */
0x0, 0x0, 0x0,
/* ipcr_mode */
0x0,
/* sflash_clk_freq */
0x0,
/* reserved_1 */
0xff,
/* reserved_2 */
0xff,
/* reserved_3 */
0xff,
/* command_sequences */
0xff, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff,
```

```
onsole [CAN_example_CT1]
 target: CAN_example_CT1.elf
: Standard S32DS C Linker
-eabi-gcc -o "CAN_example_CT1.
 building target: CAN_example_

 Standard S32DS Print Size
-eabi-size --format=berkeley C
 data     bss     dec     he
```

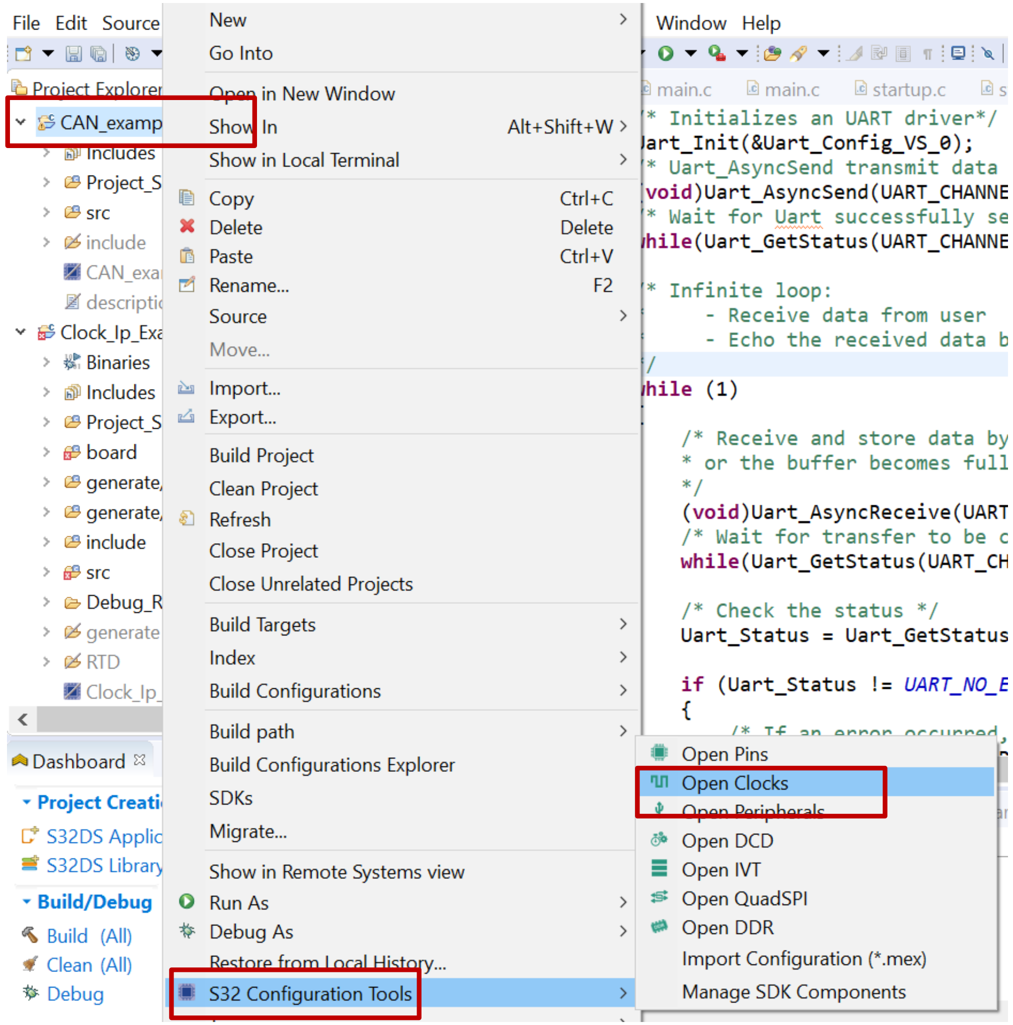




Hands on CAN: Clock Configuration 1

Go to desired configuration tool:

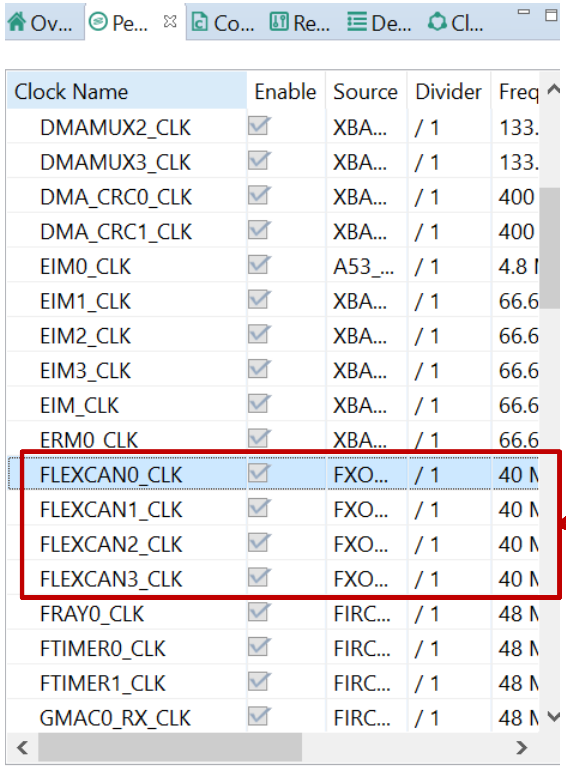
- Right click on Project,
- Select S32 Configuration Tool…
- Select Open Clocks



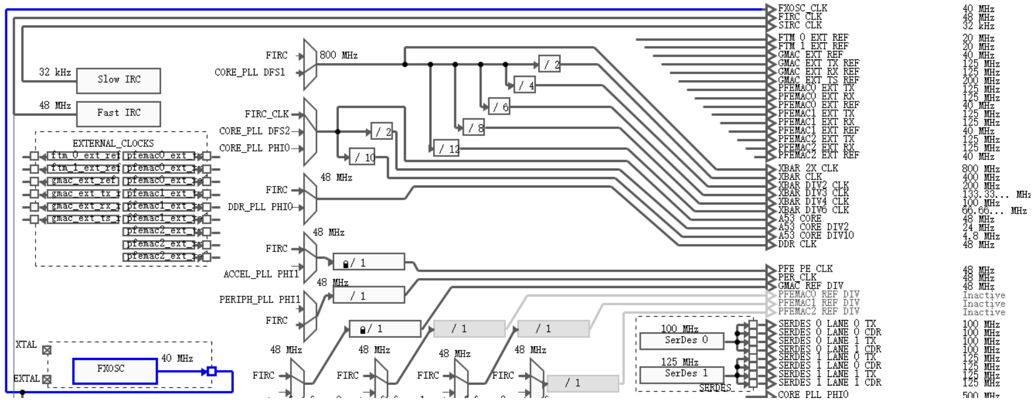
```
/* Initializes an UART driver*/
Uart_Init(&Uart_Config_VS_0);
/* Uart_AsyncSend transmit data
(void)Uart_AsyncSend(UART_CHANNE
/* Wait for Uart successfully se
while(Uart_GetStatus(UART_CHANNE

/* Infinite loop:
    - Receive data from user
    - Echo the received data b
*/
while (1)

    /* Receive and store data by
     * or the buffer becomes full
     */
    (void)Uart_AsyncReceive(UART
    /* Wait for transfer to be c
    while(Uart_GetStatus(UART_CH

    /* Check the status */
    Uart_Status = Uart_GetStatus

    if (Uart_Status != UART_NO_E
    {
        /* If an error occurred,
```

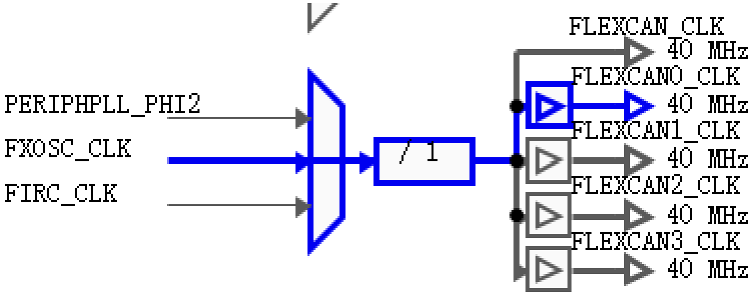
Hands on CAN: Clock Configuration 2

Open the **Peripheral Clock View**, double click the **FLEXCAN0_CLK**. The **Clocks Diagram** will show the power tree and the key node can be re-set. The default clock configuration of CAN is 40 MHZ. the CAN PE clock source comes from FXOSC



Clock Name	Enable	Source	Divider	Freq
DMAMUX2_CLK	☑	XBA...	/ 1	133.
DMAMUX3_CLK	☑	XBA...	/ 1	133.
DMA_CRC0_CLK	☑	XBA...	/ 1	400
DMA_CRC1_CLK	☑	XBA...	/ 1	400
EIM0_CLK	☑	A53_...	/ 1	4.8 I
EIM1_CLK	☑	XBA...	/ 1	66.6
EIM2_CLK	☑	XBA...	/ 1	66.6
EIM3_CLK	☑	XBA...	/ 1	66.6
EIM_CLK	☑	XBA...	/ 1	66.6
ERM0_CLK	☑	XBA...	/ 1	66.6
FLEXCAN0_CLK	☑	FXO...	/ 1	40 N
FLEXCAN1_CLK	☑	FXO...	/ 1	40 N
FLEXCAN2_CLK	☑	FXO...	/ 1	40 N
FLEXCAN3_CLK	☑	FXO...	/ 1	40 N
FRAY0_CLK	☑	FIRC...	/ 1	48 N
FTIMER0_CLK	☑	FIRC...	/ 1	48 N
FTIMER1_CLK	☑	FIRC...	/ 1	48 N
GMAC0_RX_CLK	☑	FIRC...	/ 1	48 N

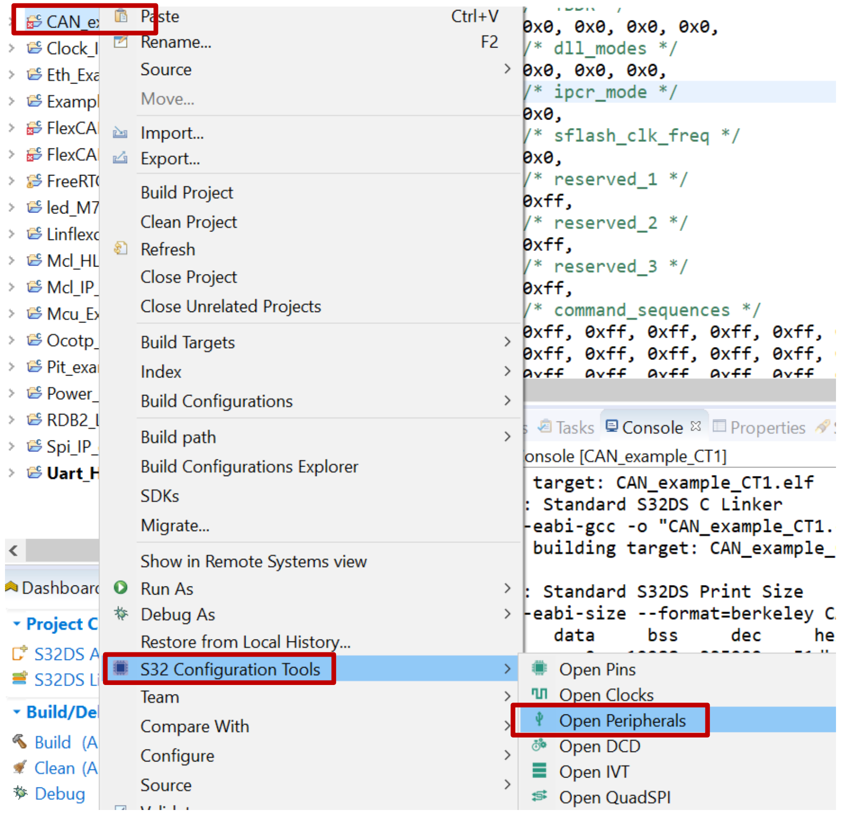




Hands on CAN: CAN Configuration 1

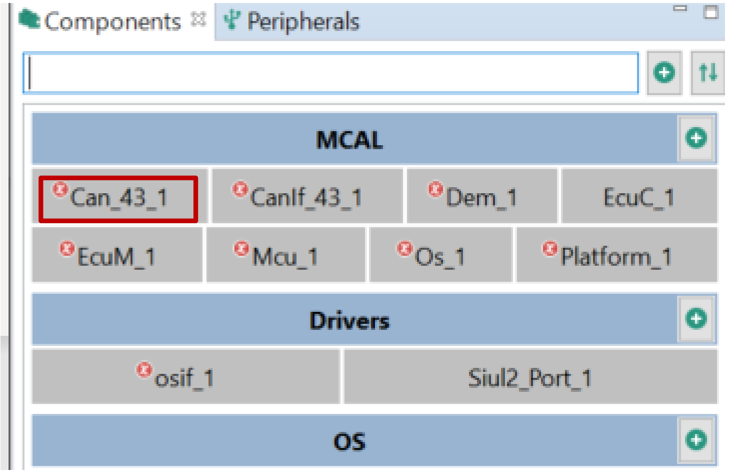
Open the Clocks Diagram:

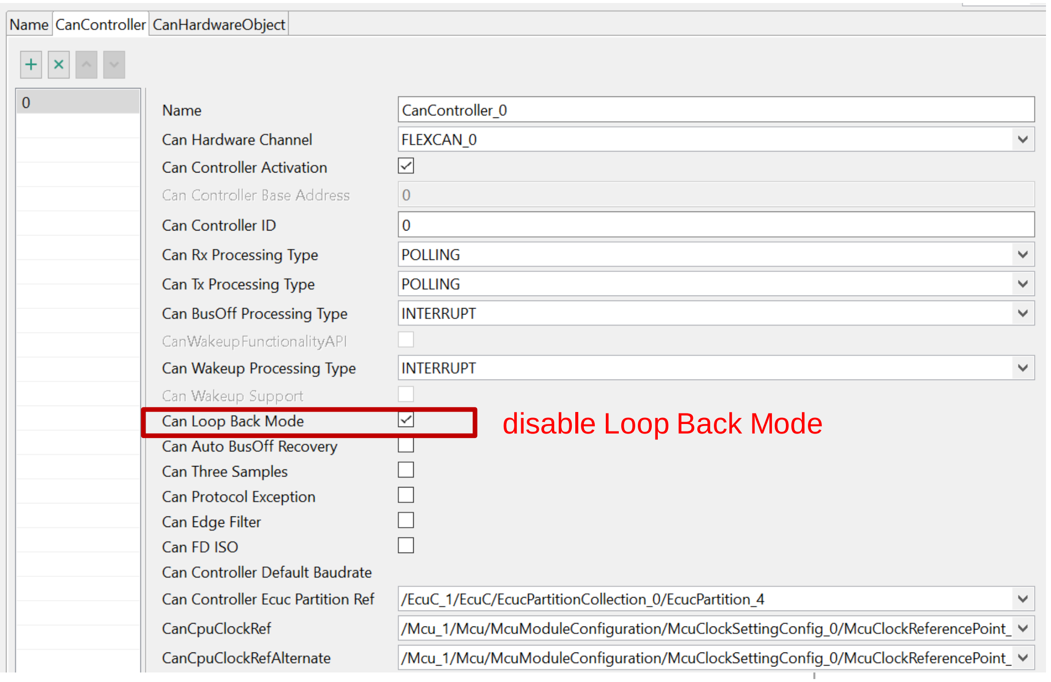
- Right click on Project,
- Select S32 Configuration Tool…
- Select Peripherals



```
0x0, 0x0, 0x0, 0x0,
/* dll_modes */
0x0, 0x0, 0x0,
/* ipcr_mode */
0x0,
/* sflash_clk_freq */
0x0,
/* reserved_1 */
0xff,
/* reserved_2 */
0xff,
/* reserved_3 */
0xff,
/* command_sequences */
0xff, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff,
```

```
onsole [CAN_example_CT1]
 target: CAN_example_CT1.elf
: Standard S32DS C Linker
-eabi-gcc -o "CAN_example_CT1.
 building target: CAN_example_

: Standard S32DS Print Size
-eabi-size --format=berkeley C
  data      bss      dec       he
```





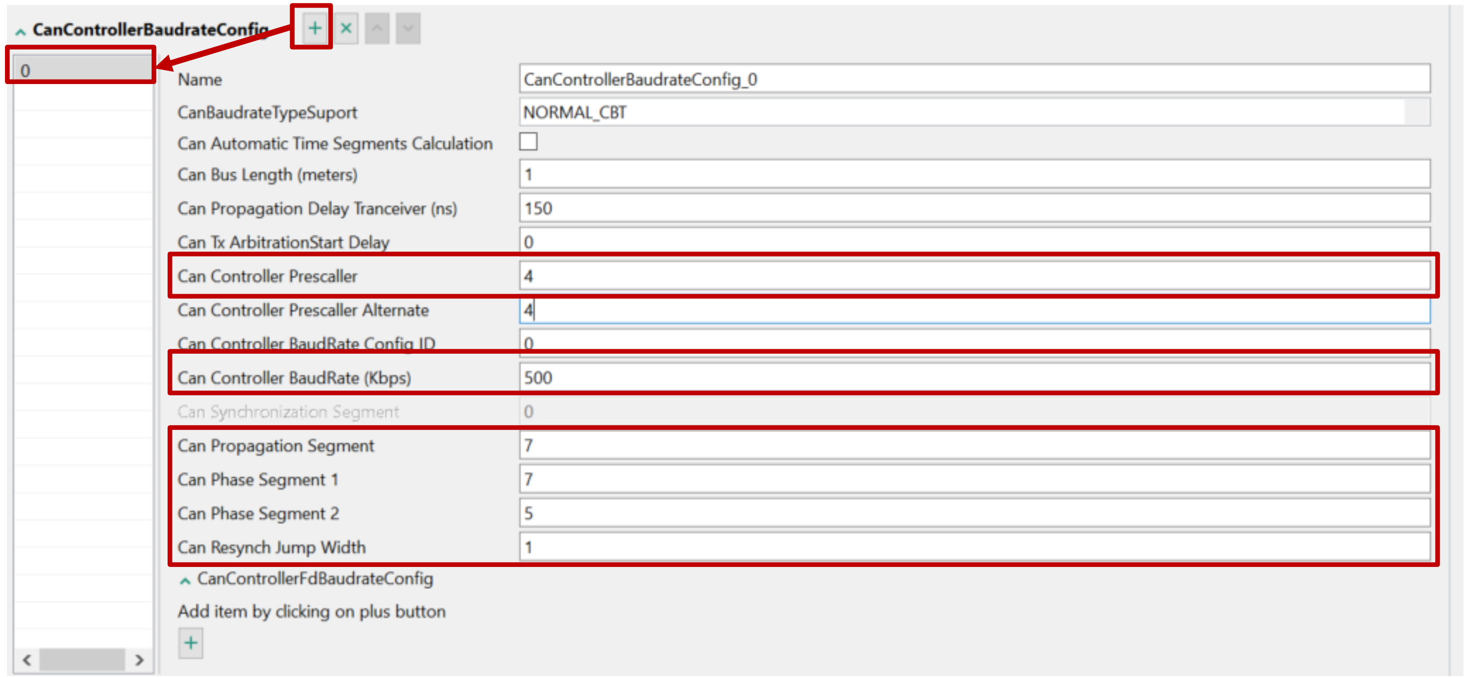
Name	CanController_0
Can Hardware Channel	FLEXCAN_0
Can Controller Activation	☑
Can Controller Base Address	0
Can Controller ID	0
Can Rx Processing Type	POLLING
Can Tx Processing Type	POLLING
Can BusOff Processing Type	INTERRUPT
CanWakeupFunctionalityAPI	☐
Can Wakeup Processing Type	INTERRUPT
Can Wakeup Support	☐
Can Loop Back Mode	☑
Can Auto BusOff Recovery	☐
Can Three Samples	☐
Can Protocol Exception	☐
Can Edge Filter	☐
Can FD ISO	☐
Can Controller Default Baudrate	
Can Controller Ecuc Partition Ref	/EcuC_1/EcuC/EcucPartitionCollection_0/EcucPartition_4
CanCpuClockRef	/Mcu_1/Mcu/McuModuleConfiguration/McuClockSettingConfig_0/McuClockReferencePoint_
CanCpuClockRefAlternate	/Mcu_1/Mcu/McuModuleConfiguration/McuClockSettingConfig_0/McuClockReferencePoint_

disable Loop Back Mode

Hands on CAN: CAN Configuration 2

Configure the Baud rate as 500Kbps for Controller 0

- TimeQuantum (seconds) = Prescaler / CanClockFrequency
- No. of CanTimeQuantas = (1 / CancontrollerBaudRate) / TimeQuantum
- No. of CanTimeQuantas = 1 + CanControllerPropSeg + CanControllerSeg1 + CanControllerSeg2


CanControllerBaudrateConfig

0

Name	CanControllerBaudrateConfig_0
CanBaudrateTypeSuport	NORMAL_CBT
Can Automatic Time Segments Calculation	☐
Can Bus Length (meters)	1
Can Propagation Delay Tranceiver (ns)	150
Can Tx ArbitrationStart Delay	0
Can Controller Prescaller	4
Can Controller Prescaller Alternate	4
Can Controller BaudRate Config ID	0
Can Controller BaudRate (Kbps)	500
Can Synchronization Segment	0
Can Propagation Segment	7
Can Phase Segment 1	7
Can Phase Segment 2	5
Can Resynch Jump Width	1

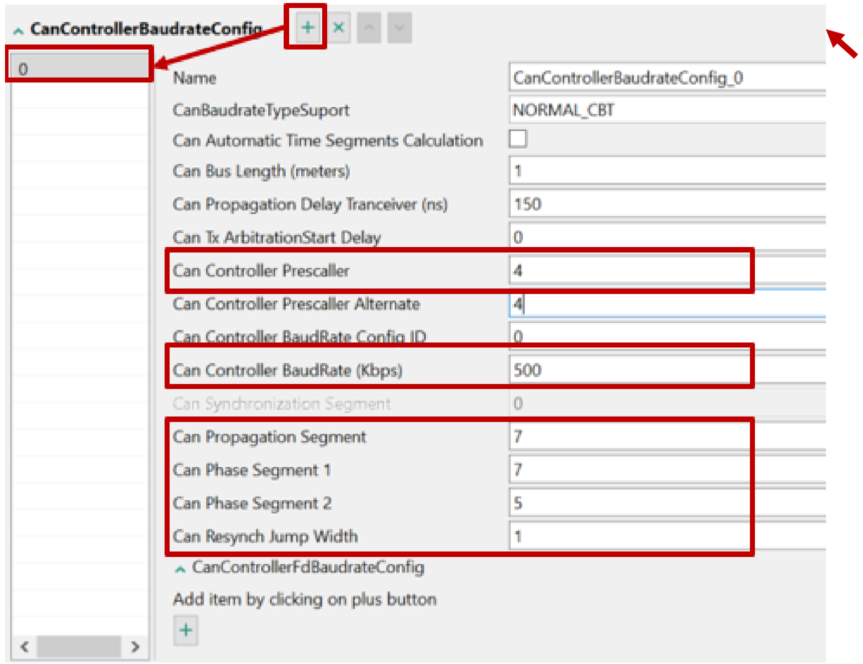
CanControllerFdBaudrateConfig

Add item by clicking on plus button

Hands on CAN: CAN Configuration 3

Open the peripheral configuration view

- add a new CanController for FLEXCAN_1
- Set Hardware Channel as FLEXCAN_1
- Set CAN Rx/TX Processing Type as POLLING
- Set CanCpuClockRef as 40Mhz
- Set Baudrate as 500kbps



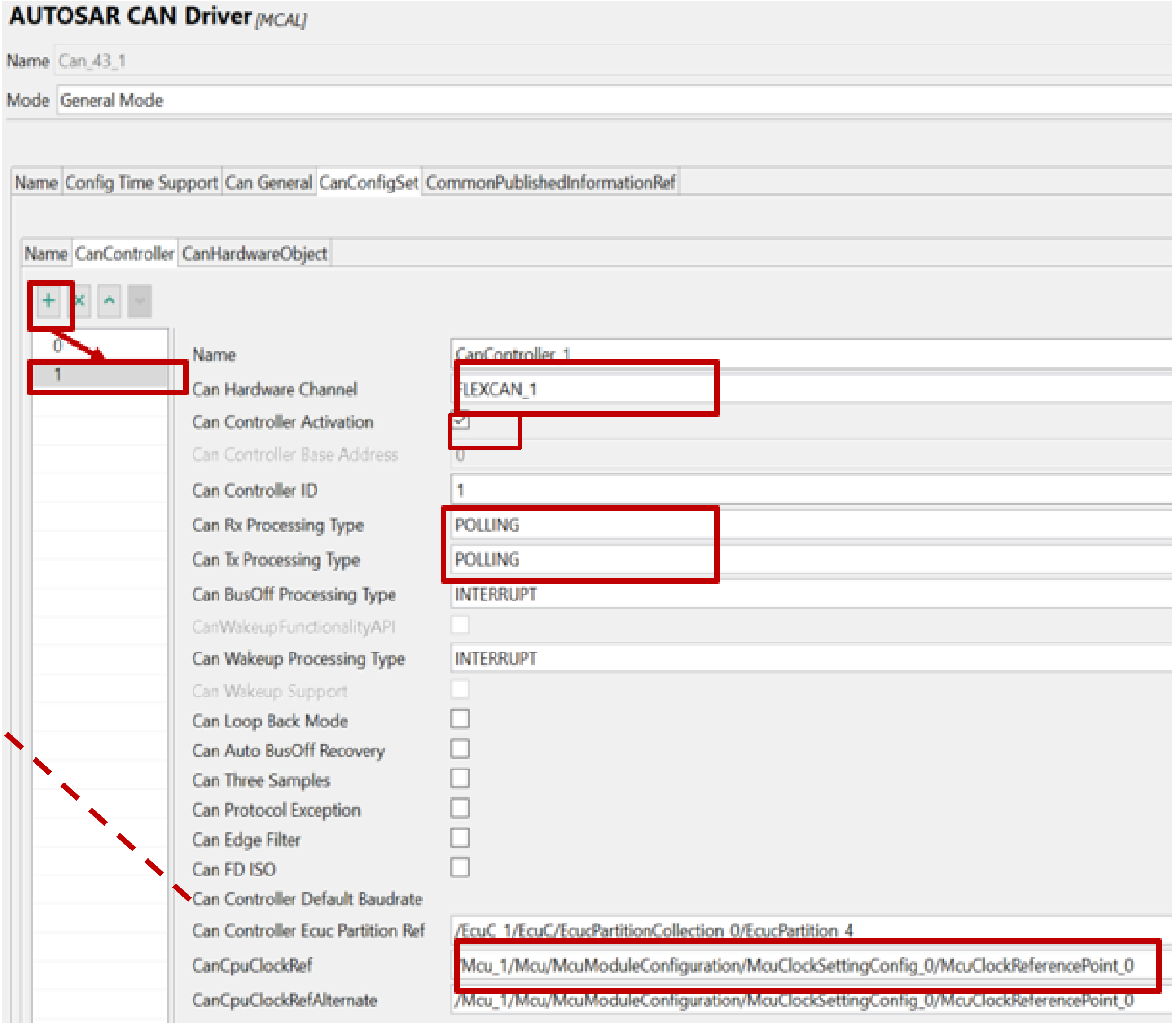
CanControllerBaudrateConfig

Name	CanControllerBaudrateConfig_0
CanBaudrateTypeSuport	NORMAL_CBT
Can Automatic Time Segments Calculation	□
Can Bus Length (meters)	1
Can Propagation Delay Tranceiver (ns)	150
Can Tx ArbitrationStart Delay	0
Can Controller Prescaller	4
Can Controller Prescaller Alternate	4
Can Controller BaudRate Config ID	0
Can Controller BaudRate (Kbps)	500
Can Synchronization Segment	0
Can Propagation Segment	7
Can Phase Segment 1	7
Can Phase Segment 2	5
Can Resynch Jump Width	1

CanControllerFdBaudrateConfig

Add item by clicking on plus button

Set Baudrate



AUTOSAR CAN Driver *[MCAL]*

Name Can_43_1

Mode General Mode

Name | Config Time Support | Can General | CanConfigSet | CommonPublishedInformationRef

Name | CanController | CanHardwareObject

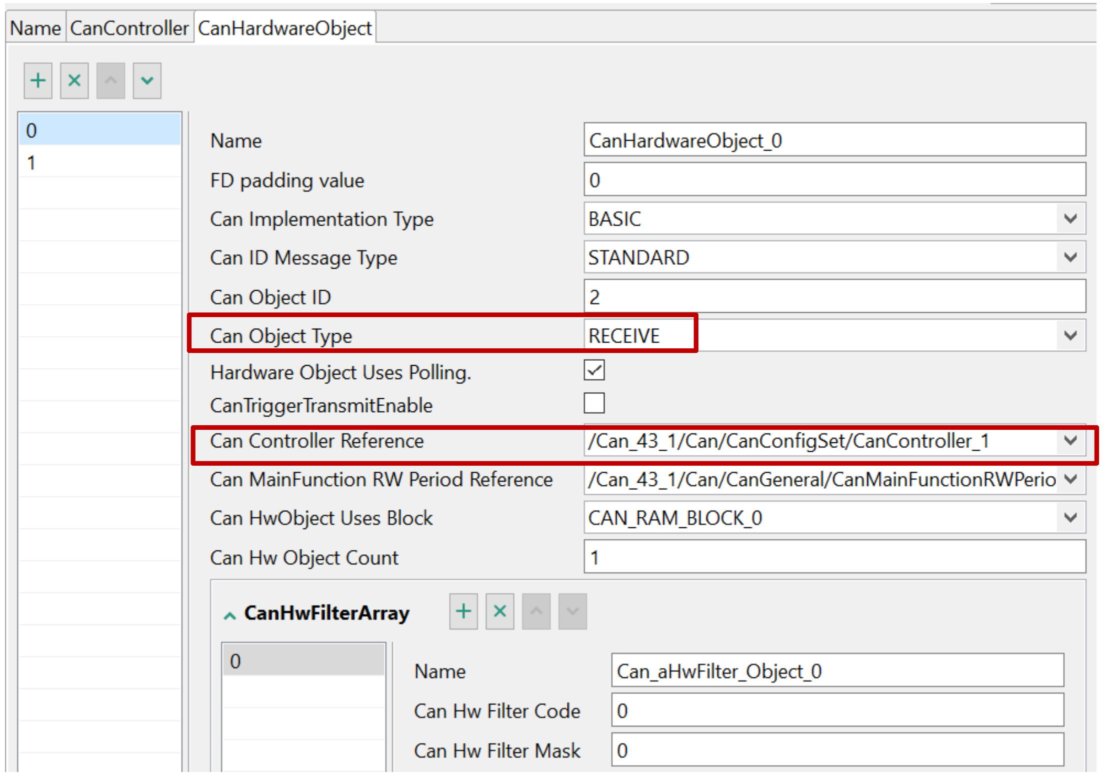
Name	CanController_1
Can Hardware Channel	FLEXCAN_1
Can Controller Activation	☑
Can Controller Base Address	0
Can Controller ID	1
Can Rx Processing Type	POLLING
Can Tx Processing Type	POLLING
Can BusOff Processing Type	INTERRUPT
CanWakeupFunctionalityAPI	□
Can Wakeup Processing Type	INTERRUPT
Can Wakeup Support	□
Can Loop Back Mode	□
Can Auto BusOff Recovery	□
Can Three Samples	□
Can Protocol Exception	□
Can Edge Filter	□
Can FD ISO	□
Can Controller Default Baudrate	
Can Controller Ecuc Partition Ref	/EcuC_1/EcuC/EcucPartitionCollection_0/EcucPartition_4
CanCpuClockRef	/Mcu_1/Mcu/McuModuleConfiguration/McuClockSettingConfig_0/McuClockReferencePoint_0
CanCpuClockRefAlternate	/Mcu_1/Mcu/McuModuleConfiguration/McuClockSettingConfig_0/McuClockReferencePoint_0

Hands on CAN: CAN Configuration 3

Modify the CanHardwareObjects Configuration for CanController 0 and CanController 1

Set the CanHardwareObjects_0 reference to CanController 1

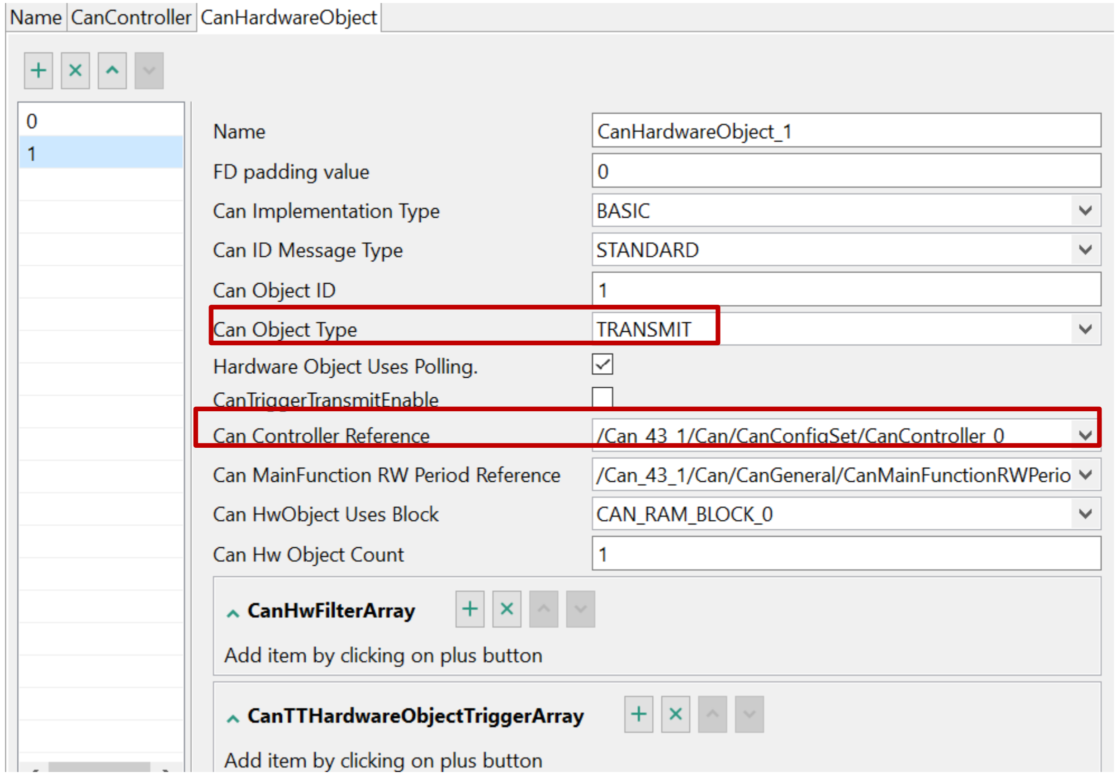
Set the CanHardwareObjects_1 reference to CanController 0



Name	CanHardwareObject_0
FD padding value	0
Can Implementation Type	BASIC
Can ID Message Type	STANDARD
Can Object ID	2
Can Object Type	RECEIVE
Hardware Object Uses Polling.	☑
CanTriggerTransmitEnable	□
Can Controller Reference	/Can_43_1/Can/CanConfigSet/CanController_1
Can MainFunction RW Period Reference	/Can_43_1/Can/CanGeneral/CanMainFunctionRWPerio
Can HwObject Uses Block	CAN_RAM_BLOCK_0
Can Hw Object Count	1

CanHwFilterArray

Name	Can_aHwFilter_Object_0
Can Hw Filter Code	0
Can Hw Filter Mask	0



Name	CanHardwareObject_1
FD padding value	0
Can Implementation Type	BASIC
Can ID Message Type	STANDARD
Can Object ID	1
Can Object Type	TRANSMIT
Hardware Object Uses Polling.	☑
CanTriggerTransmitEnable	□
Can Controller Reference	/Can_43_1/Can/CanConfigSet/CanController_0
Can MainFunction RW Period Reference	/Can_43_1/Can/CanGeneral/CanMainFunctionRWPerio
Can HwObject Uses Block	CAN_RAM_BLOCK_0
Can Hw Object Count	1

CanHwFilterArray

Add item by clicking on plus button

CanTTHardwareObjectTriggerArray

Add item by clicking on plus button

Hands on ETH: Update code

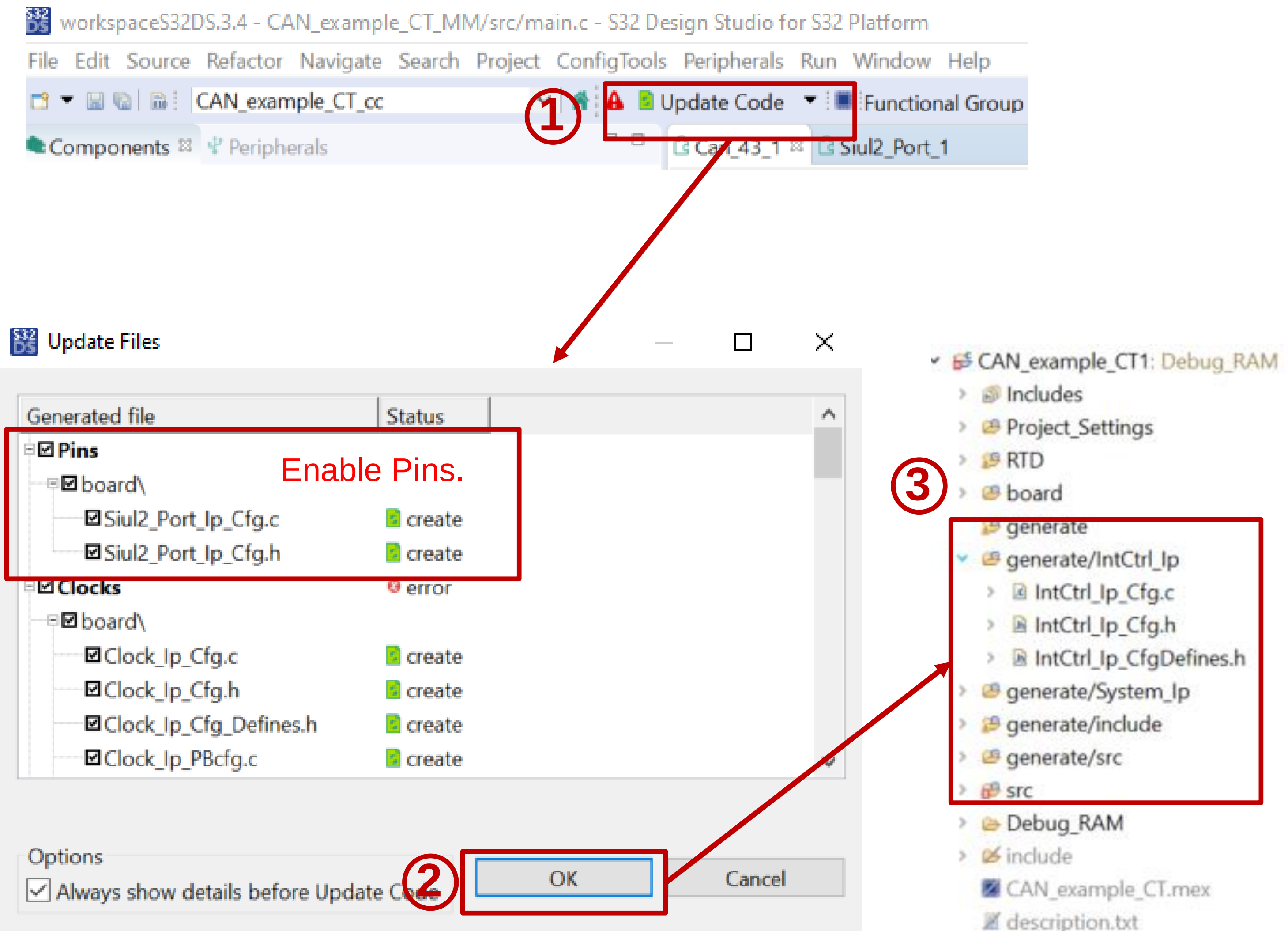
Generate code method:

1.Click on any configuration tool, like Pins

Then click **Update Code** (ensure desired project is selected!)

2.The Update Files widow pops up. It shows the detail update information. Click **ok** button.

3.The configuration .c and .h file will be generated at “generate” folder.



workspaceS32DS.3.4 - CAN_example_CT_MM/src/main.c - S32 Design Studio for S32 Platform

File Edit Source Refactor Navigate Search Project ConfigTools Peripherals Run Window Help
CAN_example_CT_cc
1
Update Code
Functional Group
Components
Peripherals
Can_43_1
Siul2_Port_1

Update Files

Generated file	Status
☑ **Pins**	
☑ board\	
☑ Siul2_Port_Ip_Cfg.c	create
☑ Siul2_Port_Ip_Cfg.h	create
☑ **Clocks**	error
☑ board\	
☑ Clock_Ip_Cfg.c	create
☑ Clock_Ip_Cfg.h	create
☑ Clock_Ip_Cfg_Defines.h	create
☑ Clock_Ip_PBcfg.c	create

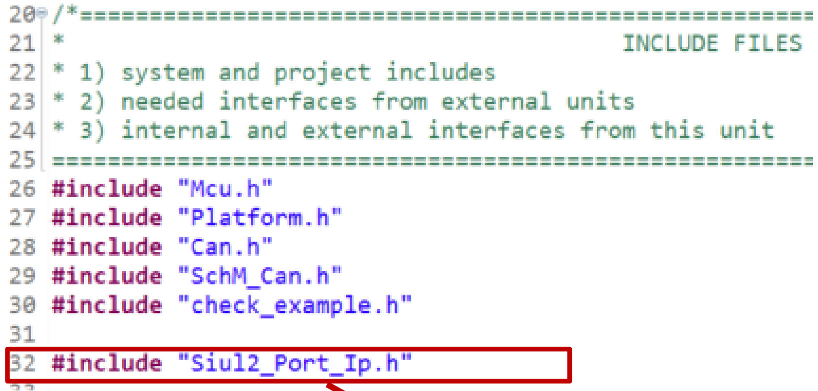
Enable Pins.

Options

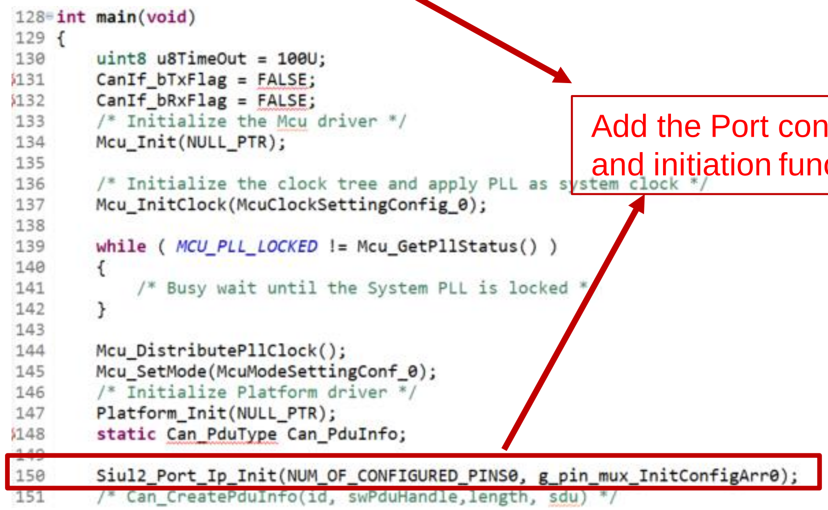
☑ Always show details before Update Code

- CAN_example_CT1: Debug_RAM
  - Includes
  - Project_Settings
  - RTD
  - board
  - generate
  - generate/IntCtrl_Ip
    - IntCtrl_Ip_Cfg.c
    - IntCtrl_Ip_Cfg.h
    - IntCtrl_Ip_CfgDefines.h
  - generate/System_Ip
  - generate/include
  - generate/src
  - src
  - Debug_RAM
  - include
  - CAN_example_CT.mex
  - description.txt

Hands on CAN: Application code



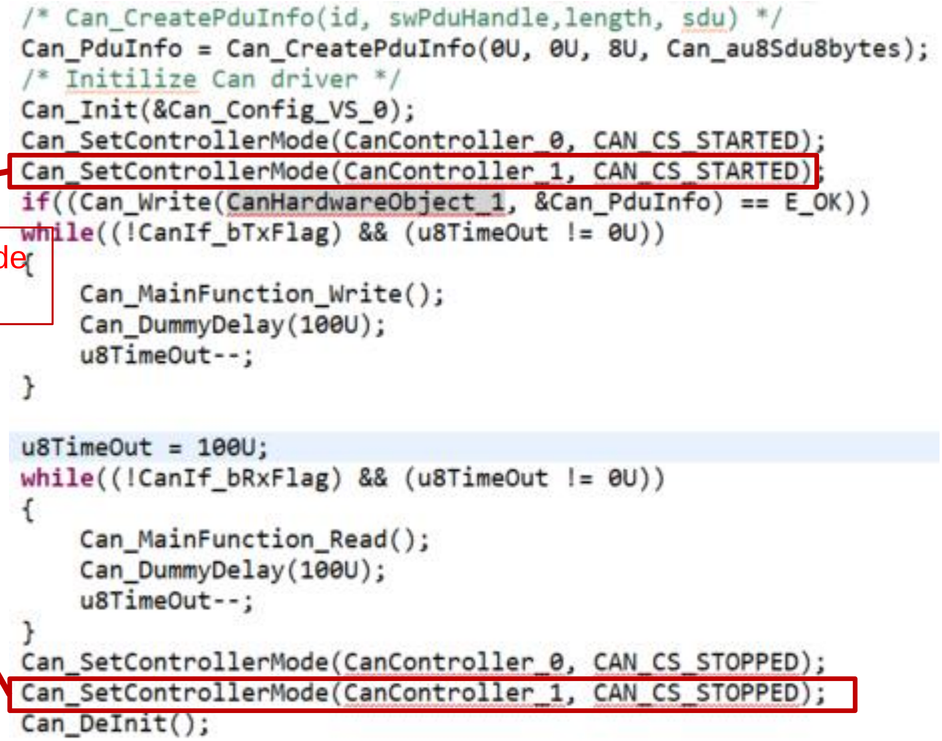
```
/*==================================================================
*                                                     INCLUDE FILES
* 1) system and project includes
* 2) needed interfaces from external units
* 3) internal and external interfaces from this unit
==================================================================
#include "Mcu.h"
#include "Platform.h"
#include "Can.h"
#include "SchM_Can.h"
#include "check_example.h"

#include "Siul2_Port_Ip.h"
```



```
int main(void)
{
    uint8 u8TimeOut = 100U;
    CanIf_bTxFlag = FALSE;
    CanIf_bRxFlag = FALSE;
    /* Initialize the Mcu driver */
    Mcu_Init(NULL_PTR);

    /* Initialize the clock tree and apply PLL as system clock */
    Mcu_InitClock(McuClockSettingConfig_0);

    while ( MCU_PLL_LOCKED != Mcu_GetPllStatus() )
    {
        /* Busy wait until the System PLL is locked */
    }

    Mcu_DistributePllClock();
    Mcu_SetMode(McuModeSettingConf_0);
    /* Initialize Platform driver */
    Platform_Init(NULL_PTR);
    static Can_PduType Can_PduInfo;

    Siul2_Port_Ip_Init(NUM_OF_CONFIGURED_PINS0, g_pin_mux_InitConfigArr0);
    /* Can_CreatePduInfo(id, swPduHandle,length, sdu) */
```

Add the Port configuration and initiation function

Add the Can_SetControllerMode for CanController_1

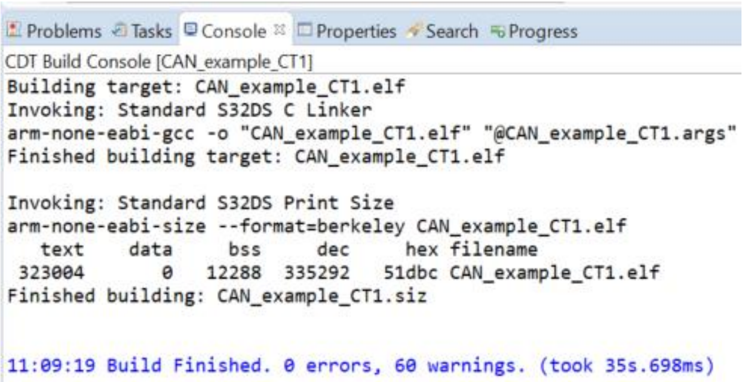


```
/* Can_CreatePduInfo(id, swPduHandle,length, sdu) */
Can_PduInfo = Can_CreatePduInfo(0U, 0U, 8U, Can_au8Sdu8bytes);
/* Initilize Can driver */
Can_Init(&Can_Config_VS_0);
Can_SetControllerMode(CanController_0, CAN_CS_STARTED);
Can_SetControllerMode(CanController_1, CAN_CS_STARTED);
if((Can_Write(CanHardwareObject_1, &Can_PduInfo) == E_OK))
while((!CanIf_bTxFlag) && (u8TimeOut != 0U))
{
    Can_MainFunction_Write();
    Can_DummyDelay(100U);
    u8TimeOut--;
}

u8TimeOut = 100U;
while((!CanIf_bRxFlag) && (u8TimeOut != 0U))
{
    Can_MainFunction_Read();
    Can_DummyDelay(100U);
    u8TimeOut--;
}
Can_SetControllerMode(CanController_0, CAN_CS_STOPPED);
Can_SetControllerMode(CanController_1, CAN_CS_STOPPED);
Can_DeInit();
```

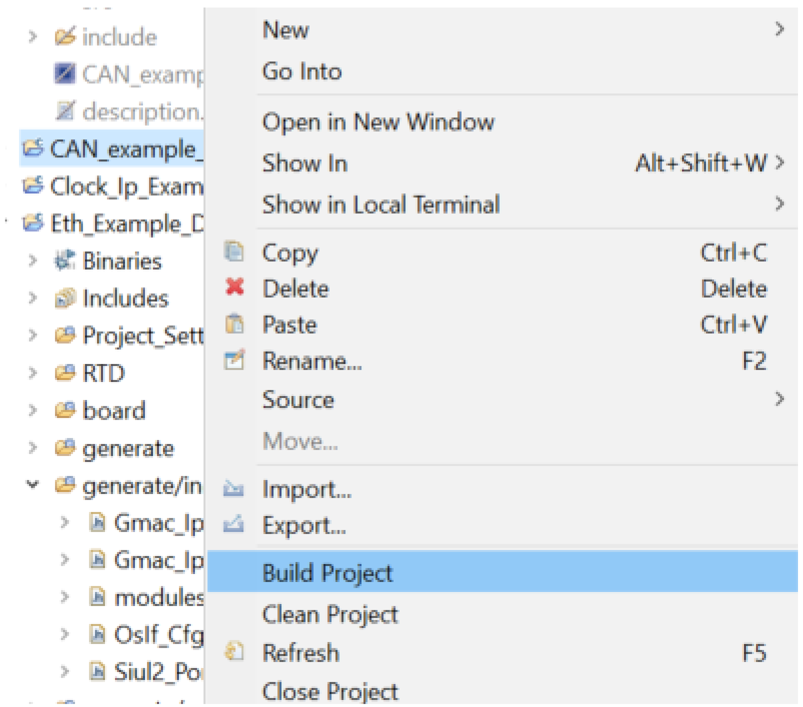
Hands on CAN: Build and Debug

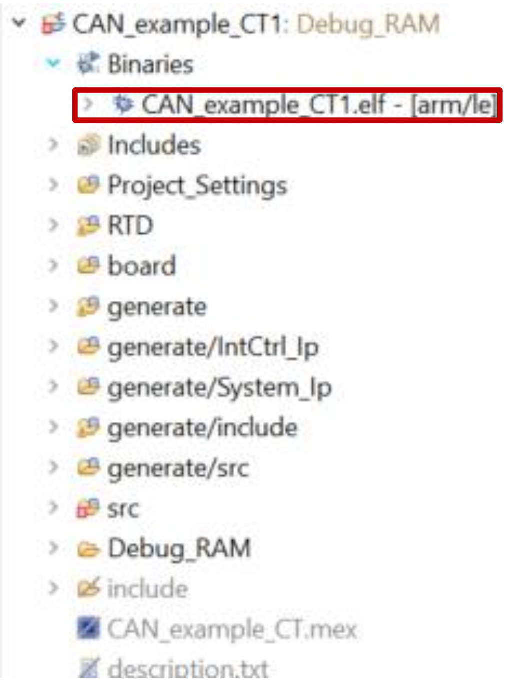
Build target Project:

- Right click on Project
- Build Project
- The console print build information
- CAN_example_CT1.elf is created



```
CDT Build Console [CAN_example_CT1]
Building target: CAN_example_CT1.elf
Invoking: Standard S32DS C Linker
arm-none-eabi-gcc -o "CAN_example_CT1.elf" "@CAN_example_CT1.args"
Finished building target: CAN_example_CT1.elf

Invoking: Standard S32DS Print Size
arm-none-eabi-size --format=berkeley CAN_example_CT1.elf
   text    data     bss     dec     hex filename
 323004       0   12288  335292   51dbc CAN_example_CT1.elf
Finished building: CAN_example_CT1.siz


11:09:19 Build Finished. 0 errors, 60 warnings. (took 35s.698ms)
```

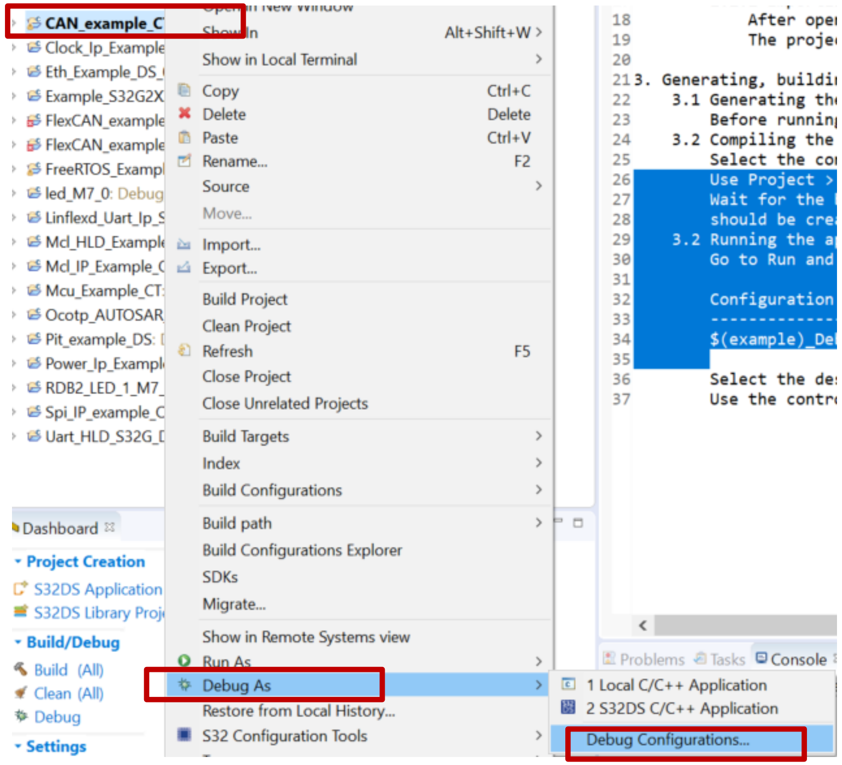




Hands on CAN: Build and Debug

Go to debug configuration:

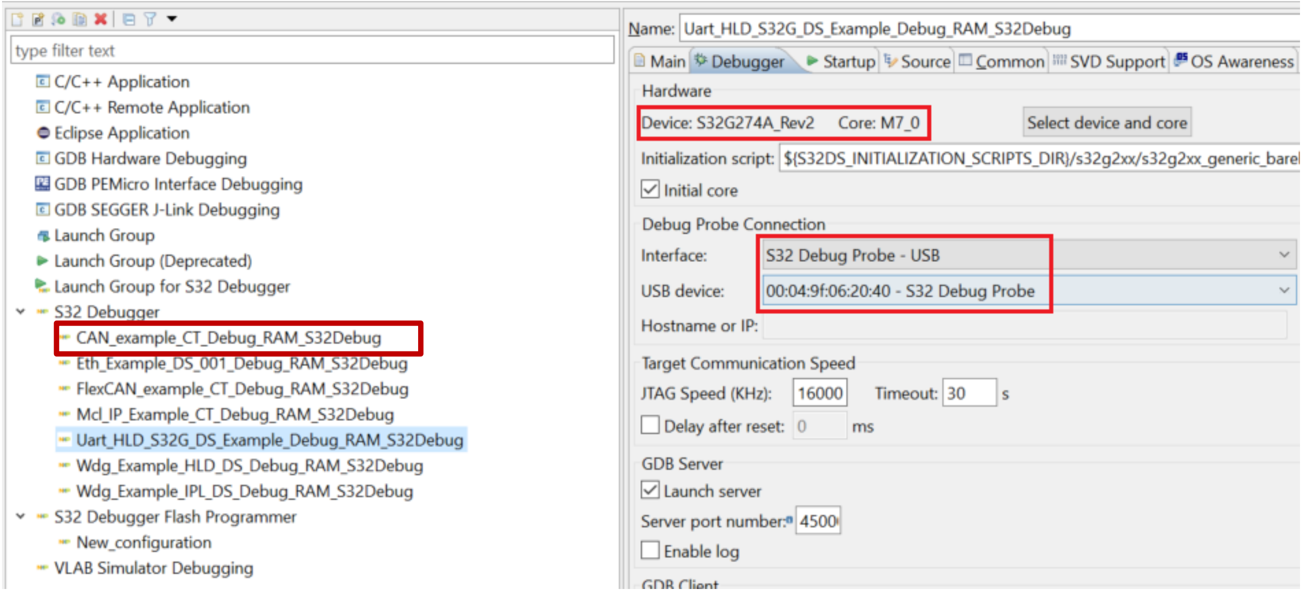
- Right click on Project,
- Select the Debug As
- Click Configurations



```
After ope
The proje

3. Generating, buildi
   3.1 Generating th
       Before runnin
   3.2 Compiling the
       Select the co
       Use Project >
       Wait for the
       should be cre
   3.2 Running the a
       Go to Run and

       Configuration
       -------------
       $(example)_De

       Select the de
       Use the contr
```

Debug configuration set:

- Click target project ,
- Select the target board
- Select target debugger


type filter text
C/C++ Application
C/C++ Remote Application
Eclipse Application
GDB Hardware Debugging
GDB PEMicro Interface Debugging
GDB SEGGER J-Link Debugging
Launch Group
Launch Group (Deprecated)
Launch Group for S32 Debugger
S32 Debugger
CAN_example_CT_Debug_RAM_S32Debug
Eth_Example_DS_001_Debug_RAM_S32Debug
FlexCAN_example_CT_Debug_RAM_S32Debug
Mcl_IP_Example_CT_Debug_RAM_S32Debug
Uart_HLD_S32G_DS_Example_Debug_RAM_S32Debug
Wdg_Example_HLD_DS_Debug_RAM_S32Debug
Wdg_Example_IPL_DS_Debug_RAM_S32Debug
S32 Debugger Flash Programmer
New_configuration
VLAB Simulator Debugging
Main
Debugger
Startup
Source
Common
SVD Support
OS Awareness
Select device and core

Name: Uart_HLD_S32G_DS_Example_Debug_RAM_S32Debug

Hardware

Device: S32G274A_Rev2 Core: M7_0

Initialization script: ${S32DS_INITIALIZATION_SCRIPTS_DIR}/s32g2xx/s32g2xx_generic_bare

☑ Initial core

Debug Probe Connection

Interface: S32 Debug Probe - USB

USB device: 00:04:9f:06:20:40 - S32 Debug Probe

Hostname or IP:

Target Communication Speed

JTAG Speed (KHz): 16000 Timeout: 30 s

☐ Delay after reset: 0 ms

GDB Server

☑ Launch server

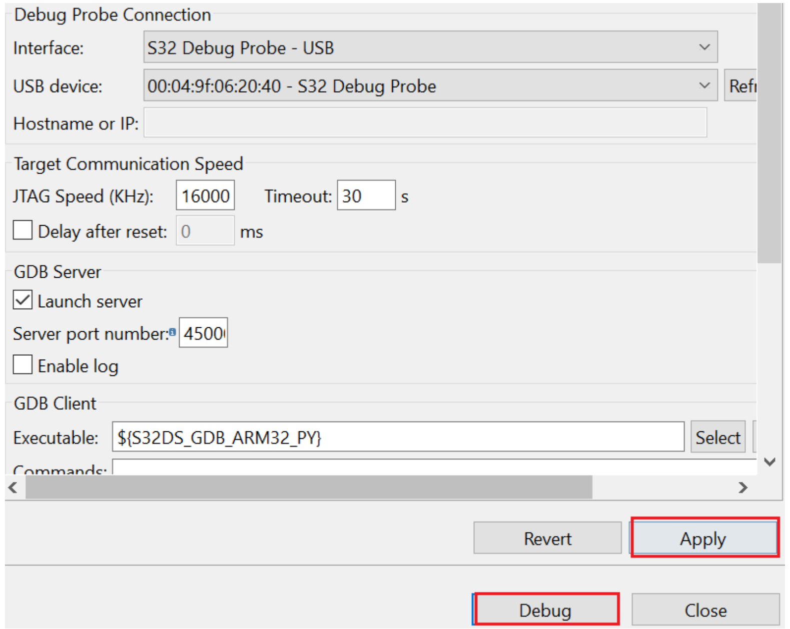
Server port number: 4500

☐ Enable log

GDB Client

Hands on CAN: Debug and run

Click on “Apply”, then click on “Debug”. the perspective will jump to the Debug Perspective, and you can use the controls to control the program flow.



Debug Probe Connection

Interface: S32 Debug Probe - USB

USB device: 00:04:9f:06:20:40 - S32 Debug Probe

Hostname or IP:

Target Communication Speed

JTAG Speed (KHz): 16000 Timeout: 30 s

□ Delay after reset: 0 ms

GDB Server

☑ Launch server

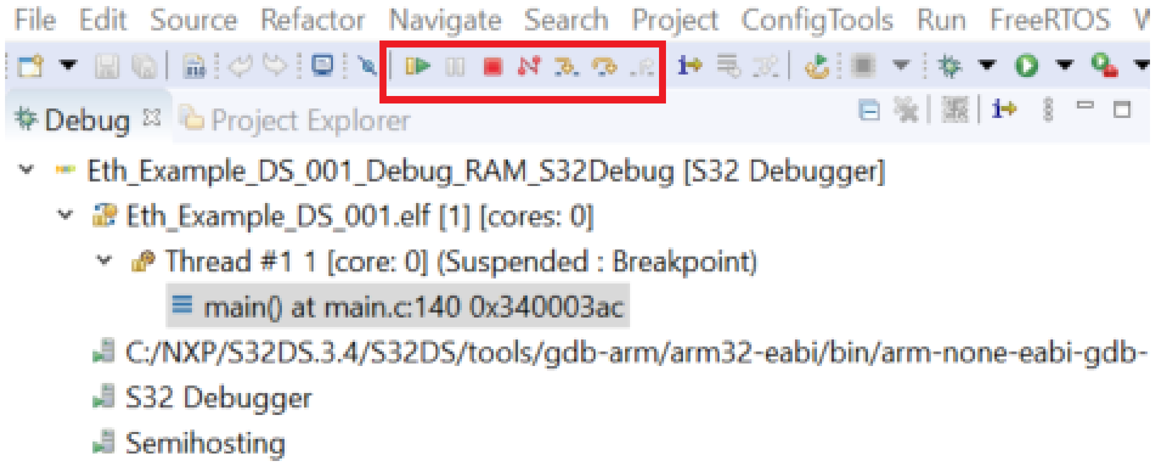
Server port number: 4500

□ Enable log

GDB Client

Executable: ${S32DS_GDB_ARM32_PY}

Commands:

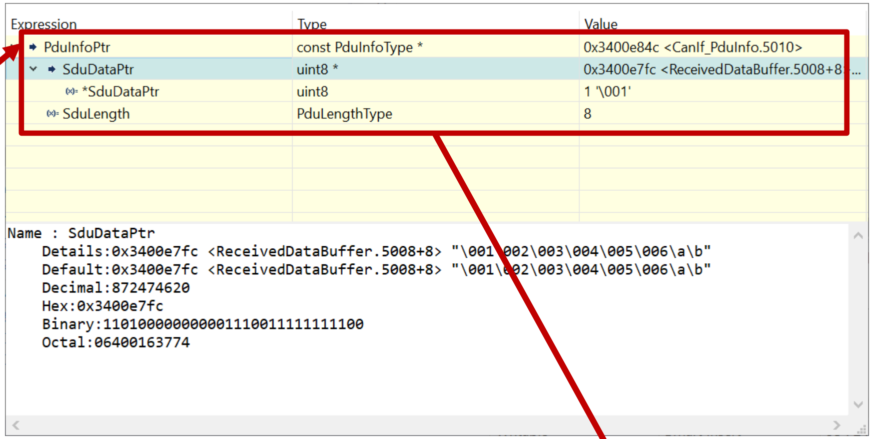


Eth_Example_DS_001_Debug_RAM_S32Debug [S32 Debugger]
Eth_Example_DS_001.elf [1] [cores: 0]
Thread #1 1 [core: 0] (Suspended : Breakpoint)
main() at main.c:140 0x340003ac
C:/NXP/S32DS.3.4/S32DS/tools/gdb-arm/arm32-eabi/bin/arm-none-eabi-gdb-
S32 Debugger
Semihosting

Hands on CAN: Test result

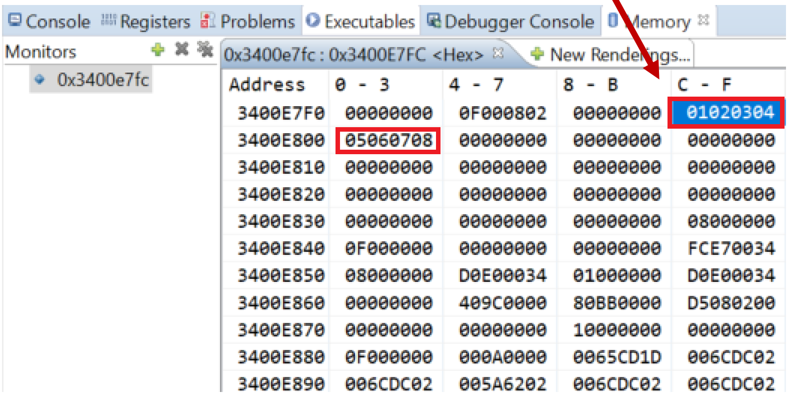
Through the modification, the CAN frame transmits from CAN0 to CAN1. the callback function **`CanIf_RxIndication`** capture the received CAN frame.

```
void CanIf_TxConfirmation(PduIdType CanTxPduId)
{
    CanIf_u8TxConfirmCnt++;
    CanIf_bTxFlag = TRUE;
}

void CanIf_RxIndication(const Can_HwType* Mailbox,
const PduInfoType* PduInfoPtr )
{
    CanIf_u8RxIndicationCnt++;
    CanIf_bRxFlag = TRUE;
}
```



Expression	Type	Value
PduInfoPtr	const PduInfoType *	0x3400e84c <CanIf_PduInfo.5010>
SduDataPtr	uint8 *	0x3400e7fc <ReceivedDataBuffer.5008+8...
*SduDataPtr	uint8	1 '\001'
SduLength	PduLengthType	8

```
Name : SduDataPtr
    Details:0x3400e7fc <ReceivedDataBuffer.5008+8> "\001\002\003\004\005\006\a\b"
    Default:0x3400e7fc <ReceivedDataBuffer.5008+8> "\001\002\003\004\005\006\a\b"
    Decimal:872474620
    Hex:0x3400e7fc
    Binary:11010000000000111001111111100
    Octal:06400163774
```



```
Address   0 - 3     4 - 7     8 - B     C - F
 3400E7F0  00000000  0F000802  00000000  01020304
 3400E800  05060708  00000000  00000000  00000000
 3400E810  00000000  00000000  00000000  00000000
 3400E820  00000000  00000000  00000000  00000000
 3400E830  00000000  00000000  00000000  08000000
 3400E840  0F000000  00000000  00000000  FCE70034
 3400E850  08000000  D0E00034  01000000  D0E00034
 3400E860  00000000  409C0000  80BB0000  D5080200
 3400E870  00000000  00000000  10000000  00000000
 3400E880  0F000000  000A0000  0065CD1D  006CDC02
 3400E890  006CDC02  005A6202  006CDC02  006CDC02
```

SECURE CONNECTIONS
FOR A SMARTER WORLD