

TMS570LS Microcontrollers: Blinky Example





Overview

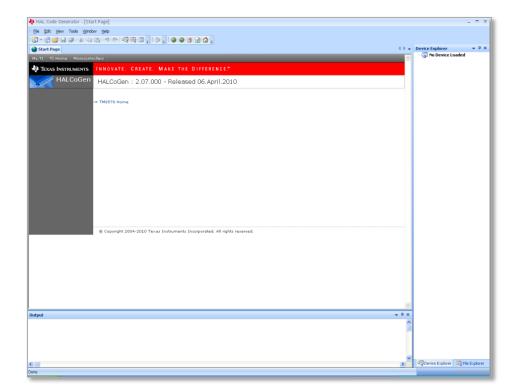
- In this example we will:
 - Create a TMS570 HALCoGen Project
 - Generate and import code into Code Composer Studio
 - Write code to blink the LED on NHET pin 1
 - Build, deploy and execute the code to the microcontroller
- Required Hardware:
 - Windows Based PC (WinXP, Vista, 7)
 - <u>TMS570LS2x USB Development Stick or Microcontroller Development Kit</u>
- Required Software:
 - <u>TMS570 HALCoGen</u>
 - <u>Code Composer Studio v4.x</u>





Setting up a New HALCoGen Project

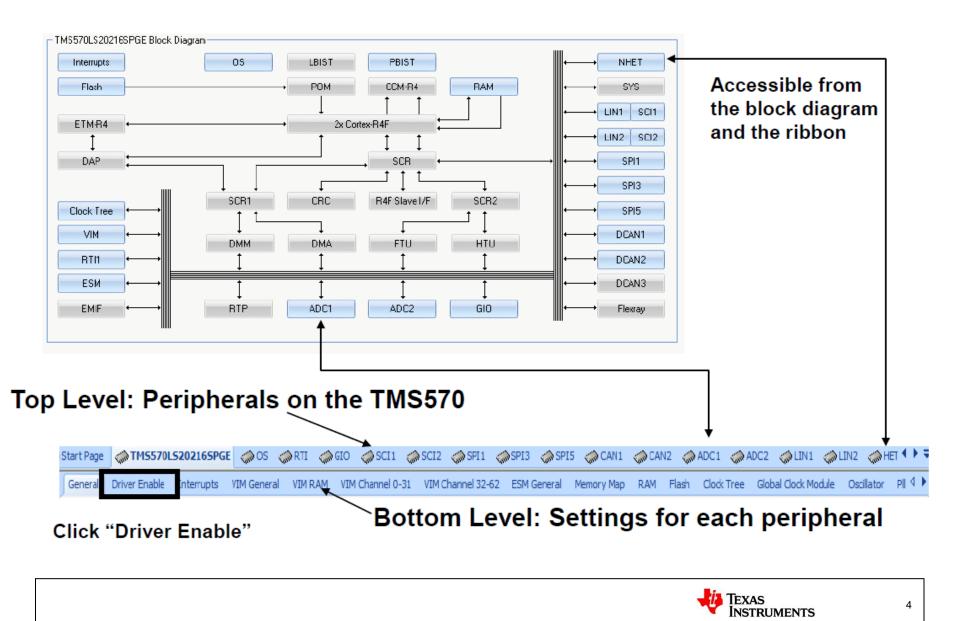
- Launch HALCoGen
 - Start \rightarrow Programs \rightarrow Texas Instruments \rightarrow HALCoGen
- File > New > Project
- Family:
 - TMDX570
- Device:
 - TMDX570LS20USB (for USB Stick) OR
 - TMDX570LS20MDK (for MDK)
- Name: TMS570 Blinky
- Location: "C:\myWorkspace"



For more help with HALCoGen, see this getting started video: LINK



The HALCoGen Interface



Configuring the Peripherals

Start Page TMS570L5202165PGE OS ORTI OGO SCI1 OSCI2 OSPI1 OSPI3 OSPI5 OCAN1 OCAN2 OADC1 OADC2 OLIN1 OLIN2 OHE T

- Select the peripherals that are required for this project.
 - In this lab we need only enable the GIO driver, uncheck all other drivers

General	Driver Enable	Cortex-R4F	Interrupts	VIM General	VIM RAM	4 🕨
🔲 En 🔽 En	Driver Compilatio able RTI driver able GIO driver able SCI driver able SPI driver	n				
Enable CAN driver Enable ADC driver						
	able LIN driver able HET driver					

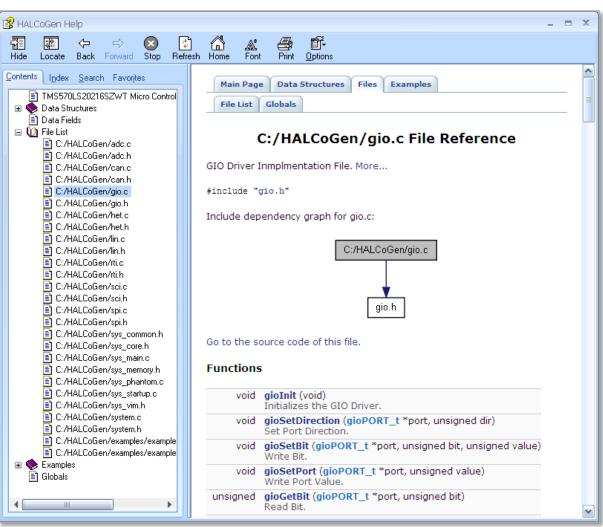
- No further changes should be made, the source code can now be generated.
 - To do this go to File \rightarrow Generate Code
 - Following, the folders on the right will populate with our new files



HALCoGen Help

- Information about the files and functions that HALCoGen creates can be found in the HALCoGen 'Help' menu
- Help can be launched from the main title bar under Help → Help Topics

🜵 HAL Code Generator - [TMS	570LS20216SPGE]
Eile Edit View Tools Win	ndow <u>H</u> elp
👬 • 👘 🐸 🖬 🖉 🐰 🖬	🖺 🤊 ୯ 💀 🖶 📮 😻 🖕 🐨 🌚 🗹 🖆
rt Page 🧼 TM5570L5202165PG	e 🗼 OS 🧼 RTI 🧼 GIO 🧼 SCI1 🧼 SCI2 🧼 SPI1



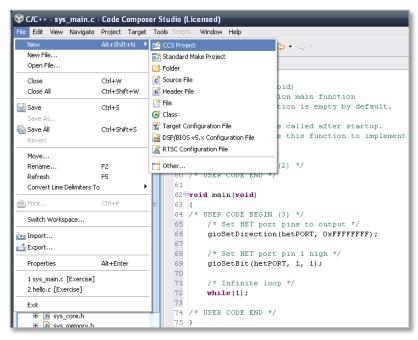


6

Setting up Code Composer Studio 4



- Launch Code Composer Studio v4.x (CCS)
 - Start \rightarrow Programs \rightarrow Texas Instruments \rightarrow Code Composer Studio v4 \rightarrow Code Composer Studio v4
- When it launches, CCS will ask you to select a workspace, we will chose "C:\myWorkspace"
- Once it loads, go to File \rightarrow New \rightarrow CCS Project



For more help with Code Composer Studio, see this getting started video: LINK



Setting up our Project

- Our project name needs to match the name of our HALCoGen Project, TMS570 Blinky
- Then Click "next"
- On the next page, make sure that your project type is set to ARM and Debug and Release are both checked
- Then Click "next"

💱 New CCS Project CCS Project		
Create a new CCS Project.		
Project name: TMS570 Blinky		
Use default location		
Location: C:/myWorkspace/TMS570 Blinky	Browse	
	🕸 New CCS Project	
	Select a type of project	-
	Select the platform and configurations you wish to deploy on	
	Project Type: ARM	~
	Configurations:	
(?) < <u>Back</u> <u>Next</u> > Einis	h Debug Se	elect All
		select All
	Show All Project Types	
	Show All Configurations	
	(2) (2)	Cancel



Setting up our Project (cont.)

• We are not using any referenced projects so click "next" again

😵 New CCS Project	$\mathbf{\overline{X}}$			
Additional Project Settings				
Define the inter-project dependencies, if any.				
Projects C/C++ Indexer				
Referenced Projects				
? < <u>B</u> ack <u>N</u> ext >	Einish Cancel			



Setting up the Project (cont.)

- Lastly, set the Device Variant to "Cortex R" and TMS570LS20216SPGE
- Click "Finish"

🕸 New CCS Project 🛛 🔀					
CCS Project Settings Select the CCS project settings.					
Output type: Executable					
Project settings					
Device Variant:	Cortex R TMS570LS202065PGE More				
Device Endianness:	big (BE32)				
Code Generation tools:	TI v4.6.1 💌 More				
Output Format:	eabi (ELF)				
Linker Command File:	Browse				
Runtime Support Library:	<automatic></automatic>				
Treat as an Assembly-	only project				
Target content None					
OUse DSP/BIOS v5.xx	✓ More				
O Enable RTSC support (required for BIOS v6.xx, Codec Engine v3.xx, etc.)					
0	< <u>Back</u> <u>N</u> ext > <u>Finish</u> Cancel				



Getting Started



- On the left hand side in the "C/C++ Projects" explorer, open "sys_main.c"
- When ever you generate code in HALCoGen, the program overwrites user code, except specific sections marked by "USER CODE BEGIN (x)" and "USER CODE END"
 - For code placement we will be referring to the number within the User Code block

/* USER CODE BEGIN (0) */ /* USER CODE END */



Writing the Code

• Inside User Code 1, copy the code below.

/* USER CODE BEGIN (1) */ #include "het.h" /* USER CODE END */

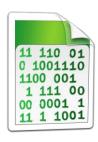




Writing the Code cont...

• Then in User Code 3, copy the code below.

```
/* USER CODE BEGIN (3) */
int temp, delay;
  /** - Delay Parameter */
  delay = 0x200000;
  /* Set HET port pins to output */
  gioSetDirection(hetPORT, 0xFFFFFFF);
  while(1)
  /* Set HET port pin 1 high */
  gioSetBit(hetPORT, 1, 1);
  /** - Simple Delay */
  for(temp=0;temp<delay;temp++);</pre>
  /* Set HET port pin 1 low */
  gioSetBit(hetPORT, 1, 0);
  /** - Simple Delay */
  for(temp=0;temp<delay;temp++);</pre>
/* USER CODE END */
```





Notifications

• Lastly we must insert a function that would be called if interrupts were enabled. Without these, the code will fail to build

```
/* USER CODE BEGIN (4) */

/* GIO Notification function not used, but required by compiler */

void gioNotification(int bit)

{

return;

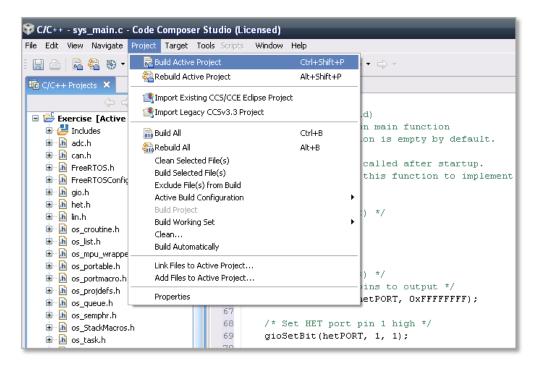
}

/* USER CODE END */
```



Compiling the Project

- The code is now complete and we are ready to build our project.
 - Go to Project \rightarrow Build Active Project



• Now that we have our .out file, we need to program the microcontrollers Flash memory.



Creating a Target Configuration

- Before we begin, we must make a new target configuration, this tells CCS4 what device this project is designed for.
 - Target \rightarrow New Target Configuration
- A new window will appear, we will make our file name "TMS570.ccxml"
- Click Finish



Creating a Target Configuration...

- A new tab will appear with a list of emulators and devices.
 - Connection: Texas Instruments XDS100v2 USB Emulator
 - In the text box labeled "Type Filter Text", type "TMS570".
 - This will narrow the search down to just TMS570 devices, select TMS570LS20216SPGE
 - Click "Save" on the right

Connection	Texas Instruments XDS100v2 USB Emulator	۴
Device	TM5570	ß
	□ TMS570LS20206SPGE □ TMS570LS20206SZWT ☑ TMS570LS20216SPGE □ TMS570LS20216SZWT	



Flash Programming Configuration

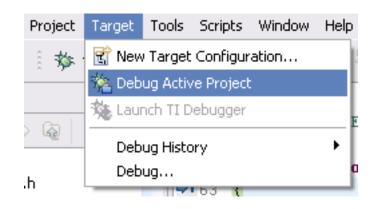
- It is possible to make the flash programming process much faster by only the necessary erasing and programming the necessary regions of flash memory.
 - To do so go to Project \rightarrow Properties
 - In the window that appears select 'CCS Debug'
 - In the CCS Debug window select the TMS570LS20216SPGE Flash Settings option in the 'Target' tab.
 - Then select the 'Necessary Sectors Only' option in the Erase Options area, then click the 'Apply' button.

	🕸 Properties for Exercis	e		- 🗆 🔀
	type filter text	CCS Debug		↓ ↓ ↓ ↓
<u>NOTE:</u> This option is only available in CCSv4.2 and newer	Info Builders C/C++ Build C/C++ Documentation C/C++ File Types C/C++ Indexer CCS Build Froject References Refactoring History		 Source Common Flash Settings Enable Programming to OTP Memory Crystal Frequency (MHz) Erase Options Entire Flash Necessary Sectors Only (for Program Load) Selected Sectors Only Bank 0 Sector 0 Sector 1 Sector 2 Sector 3 	Restore Defaults Apply
	0			OK Cancel



Programming the Flash

- We are now ready to program the flash.
 - Go to Target \rightarrow Debug Active Project
 - A new window should appear as it programs the flash memory.
 - This may take a few moments.





Testing our Program

• Click the green arrow on the debug tab to run our program



- Alternatively the program can be run without the debugger connected by
- Clicking the red square on the debug tab to terminate the debugger's connection



- Hit the reset button on the board and the NHET[1] LED should blink.
- Congratulations! You have completed the lab.



For More TMS570 Information

- TMS570 Web Page: <u>www.ti.com/TMS570</u>
 - Data Sheets
 - Technical Reference Manual
 - Application Notes
 - Software & Tools Downloads and Updates
 - Order Evaluation and Development Kits
- TMS570 Forum: <u>http://e2e.ti.com/support/microcontrollers/tms570/default.aspx</u>
 - News and Announcements
 - Useful Links
 - Ask Technical Questions
 - Search for Technical Content
- TMS570 WIKI:
 - http://processors.wiki.ti.com/index.php/Category:TMS570
 - How to guides
 - Intro Videos
 - General Information







Thank You!

For completing this TMS570 example



