

HOCHSCHULE RAVENSBURG WEINGARTEN

DEPARTMENT OF ELECTRICAL ENGINEERING AND INFORMATION TECHNOLOGY

---

# Implementation of FreeRTOS on an 8051 Softcore in an FPGA

---

*Author:*

Sohaib Saeed  
Matrikel Nr. 29422

*Contact:*

sohaibsaeed26@gmail.com

*Supervisor:*

Prof. Dr. rer. nat. Markus Pfeil

*Co-Supervisor:*

Stefan Gast M.Sc.

February 25, 2021

# Contents

<b>Abstract</b>	<b>8</b>
<b>1 Overview</b>	<b>9</b>
1.1 List of Software and Hardware used	9
1.1.1 Software	9
1.1.2 Hardware	10
1.2 Background	11
1.2.1 FreeRTOS	11
1.2.2 IP Cores	12
<b>2 Preparing the 8051 IP Softcore</b>	<b>13</b>
2.1 Creating a New Quartus Project	13
2.2 Creating the Cyclone PLL	15
2.3 Necessary changes in the Top Level Architecture	18
2.4 Pin Assignments	19
2.5 Design Compilation And Timing Analysis	20
<b>3 Debugging the FreeRTOS Cygnal Port</b>	<b>22</b>
3.1 Prerequisites	22
3.2 Changelog	23
3.2.1 Changelog #1 : c8051f120.h	23
3.2.2 Changelog #2 : portmacro.h	24
3.2.3 Changelog #3 : task.h	25
3.2.4 Changelog #4 : ParTest.c	26
3.2.5 Changelog #5 : serial.c	27
3.2.6 Changelog #6 : flash.c	29
3.2.7 Changelog #7 : semtest.c	30
3.2.8 Changelog #8 : tasks.c	31
3.2.9 Changelog #9 : queue.c	33
3.2.10 Changelog #10 : heap_1.c	34
3.2.11 Changelog #11 : port.c	35
3.2.12 Changelog #12 : main.c	36
3.2.13 Changelog #13 : Moving .rel and other files	36
3.3 packihx	37
<b>4 Adapting the Code</b>	<b>38</b>
4.1 Adapting the debugged Cygnal code for our Hardware	38
<b>5 Limitations</b>	<b>42</b>
5.1 Silicon Labs IDE	42
5.2 Testing the Debugged Cygnal Port	42
<b>6 Conclusion</b>	<b>45</b>
<b>Bibliography</b>	<b>46</b>

# List of Figures

1.1	Task Execution with Scheduling [11]	11
1.2	Oregano Systems 8051 IP Core Block Diagram	12
2.1	Quartus New Project Wizard, Directory selection	13
2.2	Quartus New Project Wizard, Adding Project files	14
2.3	Quartus New Project Wizard, Selecting the CYC1000 development board	14
2.4	Creating the Cyclone PLL, ATPLL	15
2.5	Creating the Cyclone PLL, selecting file directory	15
2.6	Creating the Cyclone PLL, Selecting the Input Frequency	16
2.7	Creating the Cyclone PLL, Deselecting Asynchronous Reset	16
2.8	Creating the Cyclone PLL, Setting Output Clock Frequency	16
2.9	Creating the Cyclone PLL, Summary	17
2.10	Creating the Cyclone PLL, Checking the <i>cyclonepll.qip</i> file	17
2.11	Oregano Systems 8051 IP Core Synthesis Errors	18
2.12	Oregano Systems 8051 IP Core RTL View	18
2.13	Oregano Systems 8051 IP Core, replaced line 154	18
2.14	Timing Constraints - Critical Warnings	20
2.15	Creating the SDC file	21
2.16	Synthesizing after implementing an SDC file	21
3.1	Extracting the FreeRTOS source code	22
3.2	GNU 'make' utility file placement	22
3.3	Syntax error, declaration ignored at 'sfr', Syntax error, declaration ignored at 'sbit'	23
3.4	syntax error: token -> 'interrupt'	24
3.5	syntax error: token -> 'push' ;	24
3.6	error 226: no type specifier for 'xTaskCreate parameter 6'	25
3.7	"const" keyword removed from line 345 (task.h)	25
3.8	error 98: conflict with previous declaration of 'vParTestSetLED' for attribute 'type'	26
3.9	warning 84: 'auto' variable 'ucBit' may be used before initialization	26
3.10	"unsigned char ucBit" added to line 46 (ParTest.c)	27
3.11	Syntax error, declaration ignored at 'data'	27
3.12	error 98: conflict with previous declaration of 'xSerialGetChar' for attribute 'type'	28
3.13	"signed" keyword added to line 148 and 166 (serial.c)	28
3.14	error 78: incompatible types	29
3.15	commented out include for "print.h" and included "queue.h" (flash.c)	29
3.16	copy function "vPrintDisplayMessage" (print.c)	29
3.17	paste function "vPrintDisplayMessage" and rename to "my_vPrintDisplayMessage", and edit function call (flash.c)	30
3.18	warning 112: function 'vPrintDisplayMessage' implicit declaration, error 101: too many parameters	30
3.19	commented out include for "print.h" and included "queue.h" (semtest.c)	31
3.20	paste function "vPrintDisplayMessage" as is, without changing anything else (semtest.c)	31
3.21	error 226: no type specifier for 'prvInitialiseNewTask parameter 6'	31
3.22	"const" keyword removed from line 548 (tasks.c)	32
3.23	"const" keyword removed from line 731 (tasks.c)	32
3.24	"const" keyword removed from line 822 (tasks.c)	32
3.25	warning 196: pointer target lost const qualifier	33

3.26	"const" keyword added to line 1997 (queue.c) . . . . .	33
3.27	"const" keyword added to line 2014 (queue.c) . . . . .	33
3.28	"const" keyword added to line 2371 (queue.c) . . . . .	34
3.29	"const" keyword added to line 2412 (queue.c) . . . . .	34
3.30	warning 127: non-pointer type cast to generic pointer . . . . .	34
3.31	Illegal pointer typecast operation in line 91 (heap_1.c) . . . . .	35
3.32	error 1: Syntax error, declaration ignored at 'data' . . . . .	35
3.33	syntax error: token -> 'void' . . . . .	36
3.34	?ASlink-Error-<cannot open> : "ParTest/ParTest.rel" . . . . .	36
3.35	Using packihx to convert the .ihx file to a .hex file [13] . . . . .	37
4.1	Pinout for the EFM8BB1 [15] (standard 8051 architecture) . . . . .	38
4.2	P3 outputs disabled and P1 outputs enabled (ParTest.c) . . . . .	39
4.3	P3 outputs replaced by P1 outputs (1) (ParTest.c) . . . . .	39
4.4	P3 outputs replaced by P1 outputs (2) (ParTest.c) . . . . .	40
4.5	Define the bit position relative to Port 2 as "my_ ..." variables (main.c) . . . . .	40
4.6	Enable Port 2 outputs as Push-Pull (main.c) . . . . .	40
4.7	Setting outputs of "prvToggleOnBoardLED" function to P2 (main.c) . . . . .	41
5.1	Using the Silicon Labs Flash Programming Utility . . . . .	42
5.2	Silicon Labs Flash Programming Utility Error message . . . . .	43
5.3	EDSIM51 MOVX warning message . . . . .	43

# List of Tables

2.1	Oregano Systems 8051 IP Core Pin Assignments . . . . .	20
3.1	Source files and their respective directories . . . . .	37

# Declaration of Authorship

I, Sohaib Saeed declare that this Bachelor Thesis, titled **Implementation of FreeRTOS on an 8051 Softcore in an FPGA** and the work presented in this report is my own. I confirm that:

- This work was done wholly or mainly while in candidature for an engineering degree at this University.
- Where any part of this project and report has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this project is entirely my own work.
- I have acknowledged all main sources of help.
- Where this project is based on work done by me jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:



---

Sohaib Saeed, 29422

# Acknowledgements

I would like to thank my supervisors, Markus Pfeil and Stefan Gast for giving me the opportunity and guidance to work on this project. They were always present for any questions or problems I had during this semester, and helped provide insightful information and advice to help me finish this thesis report. I would also like to thank my University - Hochschule Ravensburg Weingarten for accepting my application to this Bachelors program and giving me the opportunity to travel abroad and discover a world beyond my borders.

Additionally I would like to thank my friends for helping me complete my degree and for the love and support they provided constantly. I would not be here if it was not for their input in my daily life.

# Abstract

With the current rapid developments in Embedded Systems Engineering and Automation Technology, the use of real-time operating systems is becoming exceedingly common. One of the most well known and capable of these operating systems is "*FreeRTOS*". FreeRTOS allows users to define multiple recurring tasks with fixed (or variable) priorities. These tasks are then presented to the FreeRTOS scheduler which makes sure that every task is called within its given deadlines and that higher priority tasks always get called in favor of the lower priority tasks if there is ever a conflict.

Unfortunately, despite all the support and resources available for FreeRTOS, it is indeed possible for users to run into unexpected errors while trying to use a real-time kernel. Errors such as priority inversions, missed task deadlines, and scheduler faults are difficult to recreate and even harder to solve. To help with such issues, it is my Thesis proposal to try to run FreeRTOS on a well known microcontroller (Intel 8051) which has been implemented entirely in software, on an FPGA.

The advantage of using an FPGA to run the 8051 IP Core is that it we gain the ability to delve deeper into hardware and truly be able to see tasks running in real-time using various debugging signals. Using an FPGA also allows us to be quite flexible when handling signals or external peripherals related to the microcontroller, it is even possible for us to modify the architecture of the softcore if necessary, which allows us to be able to debug issues much easier than before. Additionally I chose the 8051 microcontroller as it has been rigorously tested throughout the years and boasts an extensive range of options for debugging and support online.



# Chapter 1

## Overview

### 1.1 List of Software and Hardware used

In this Thesis, I hope to present a clear guide on how to get an 8051 microcontroller softcore to run FreeRTOS and use internal signals within the FPGA to track tasks that have been scheduled. I worked on a Windows 10 64-Bit Operating System, every software and hardware I required is listed below, along with citations that specify the download links. The finalized code and project files have also been uploaded to a Git Repository [1].

#### 1.1.1 Software

##### FreeRTOS Source Code

The FreeRTOS real-time kernel source code [2] contains a pre-configured demo application known as "*Cygnal*" for the 8051 microcontroller. This demo application uses the "*Small Device C Compiler*" (SDCC) and needs to be modified for use on an IP Softcore. Additionally, the FreeRTOS website provides a useful quickstart guide for those interested [3].

##### Oregano Systems 8051 IP Core

The Oregano Systems 8051 IP Core [4] has been developed with the help of the **Vienna University of Technology** and is entirely compatible with the 8051 processor from Intel. It is a free of charge IP Softcore that I considered most suitable to be used for my thesis as it is a close match to the original 8051 and is well documented.

##### Quartus Prime Lite (Free Edition) Release 20.1.1

The Quartus Prime Lite Software [5] is an excellent tool to use for analysis and synthesis of HDL Designs as it offers a very easy to use user interface for code optimization and debugging. The Lite version of this software is free to use and has support for Cyclone 10 devices.

##### Small Device C Compiler (SDCC) v4.0.0

The SDCC [6] was required to compile the code as the Cygnal Port was designed to work with SDCC keywords and preprocessor statements. However it should be noted that I used version 4.0.0, whereas the Cygnal Port was developed using a version older than 2.4.0. Unfortunately, sometime before this version, but not before the Cygnal Port was developed; The SDCC was updated to use slightly different keywords. This meant that the original code written for the Cygnal Port could not be compiled "as-is" without errors and thus needed to be modified to work with the latest versions of the SDCC.

##### GNU 'make' Utility

GNU 'Make' Utility [7] was required to use the Makefile provided within the Cygnal Port to compile the code.

### 1.1.2 Hardware

#### Cyclone 10 CYC1000 FPGA

In terms of hardware - a relatively cheap FPGA from the Cyclone 10 series was used, namely the CYC1000 [8]. This board comes with the following key features:

- Intel Cyclone 10 LP FPGA : 25 kLE, 66 Memory blocks (9K), 594 Memory blocks (Kb), 18 x 18 multipliers -> 66
- 8 MByte SDRAM
- 2 MByte Flash (EPCQ16A)
- LIS3DH 3-axis Sensor
- 21 I/O Arduino MKR compatible headers
- Pmod: 2x6 Pin support
- 8 x user LEDs
- 1 x user push button
- 5.0 V single power supply with on-board voltage regulators

#### 8051 Development Kit

Additionally, I used a Busy Bee 8051 Development Kit [9] to first test the Cygnal Port's functionality on a real 8051 hardware architecture before porting it to the 8051 IP Softcore to eliminate further sources of possible errors.

## 1.2 Background

This section of the report aims to give a basic outline and description of the key concepts that are partially prerequisite to the core content of the report. It may not be necessary to fully understand these concepts, however they provide a valuable knowledge-base as a foundation and allow for easier debugging and troubleshooting when required.

### 1.2.1 FreeRTOS

"FreeRTOS" is defined as a Free, high Quality Real-time Operating System (RTOS) [10]. It allows users to deterministically (i.e. predictably) run multiple programs at the same time. The "*Scheduler*" present within an Operating System (OS) is responsible for deciding which program to run and at what time to ensure that any real-time requirements are met. A "*real-time requirement*" can be described as a deadline to be met when the microcontroller responds to a certain event. A timing requirement whose consequences can be the difference between life or death or extreme business losses are considered to be "*hard*" real-time requirements. (*e.g.*: The airbag deployment in case of a car crash, or the systems tracking a patient's health in a Hospital). Alternatively, "*soft*" real-time requirements are mainly those deadlines which, if missed, can be a problem, but do not pose an extreme threat. A guarantee that deadlines will be met is only possible if our system is deterministic.

The way FreeRTOS's scheduler remains deterministic is by asking the programmer to assign priorities to the tasks that need to be performed, and using this information automatically alots processing time to the tasks based on their priority. It is possible to communicate between tasks and it is also possible to block or resume tasks based on the user's needs.

FreeRTOS itself is designed for use in microcontrollers which are often very limited in size and memory. The limitations almost never allow for a full implementation of FreeRTOS and commonly only make use of the very core or the FreeRTOS "*kernel*". Each program or task defined by the user under FreeRTOS is executing through the control of the operating system or scheduler. If there are multiple programs that seem to run concurrently, it is known as "*multitasking*".

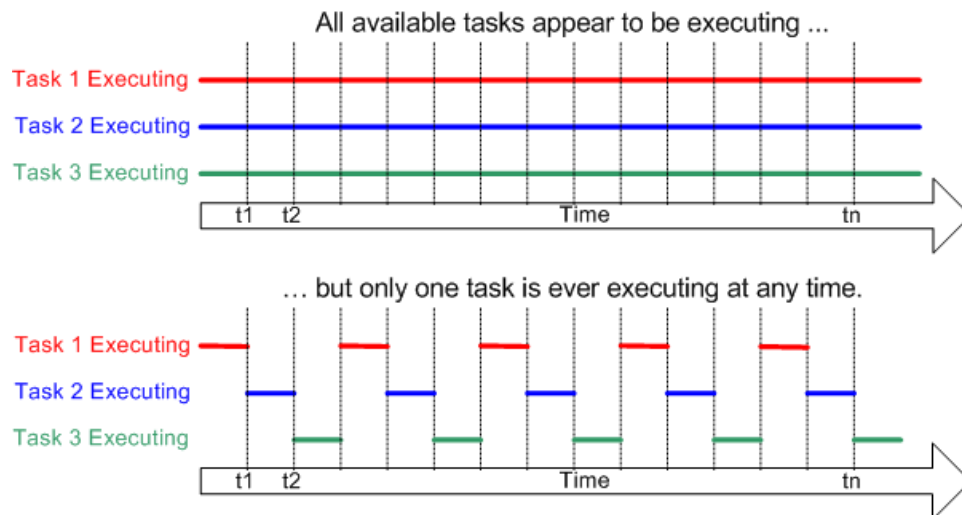


Figure 1.1: Task Execution with Scheduling [11]

### 1.2.2 IP Cores

An "IP Core" is defined as a prefabricated block of a chip design which is the "Intellectual Property" of the developer or manufacturer. In the case of the 8051, it is the Intellectual Property of **Intel**, however the license for the architecture and block of chip design has been bought by many chip manufacturers and designers across the world. **Oregano Systems** is one of these chip designers that has produced, through a hardware descriptive language such as VHDL, a purely software implemented version (also known as a "soft IP Core") for the 8051 that can run on an FPGA. The source code for the Oregano Systems 8051 IP Softcore is free to download, use and implement in any way and as such was an excellent starting point for the purposes of my thesis.

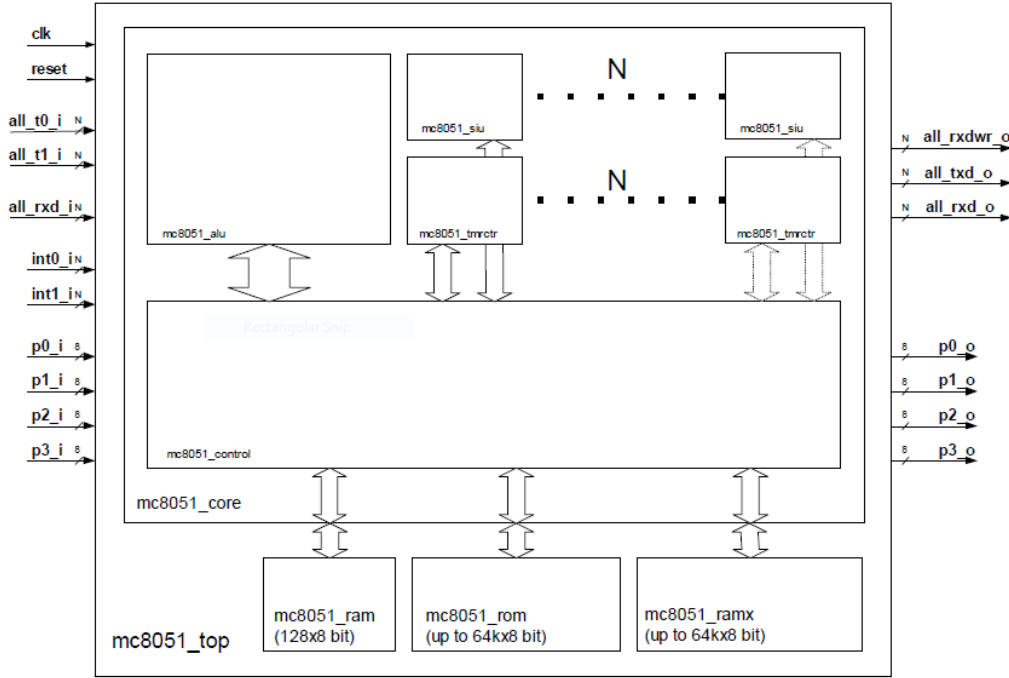


Figure 1.2: Oregano Systems 8051 IP Core Block Diagram

Additionally, with the help of extensive documentation resources available for the Oregano Systems 8051 IP Softcore, it was also easily possible to adapt the source code according to my needs in regards to the FreeRTOS Cygnal Port. As stated on the download page [4], The IP Core features:

- Fully synchronous circuit design
- Single clock
- Synthesizable circuit description
- OP code compatible to original Intel 8051 device
- Faster command execution due to a new architecture
- Separate RAM and ROM data/address bus
- Parameterizable number of timer and UART units
- Conventional multiplying algorithm or fast parallel multiplier units
- Conventional dividing algorithm or fast parallel divider units

## Chapter 2

# Preparing the 8051 IP Softcore

### 2.1 Creating a New Quartus Project

The original source code for this project comes from the Oregano Systems website [4] where I downloaded the 8051 IP Core Boot Loader Demo Design. The original Quartus project file downloaded within the ZIP file was initially intended for the EP1C20 series family of FPGAs and to reduce conflicts between the old project file settings and the new project file that needs to be created, I opted to create a new project from scratch which would contain only the necessary VHDL source files.

From the directory `"../mc8051_boot/vhdl"`, copy the folder named `"vhdl"` and all files contained within except the file named `"mc8051_top_struc_cfg.vhd"` into the directory where the new project file will be created. Additionally, the files `"mc8051_pram.vhd"`, `"mc8051_ram.vhd"`, `"mc8051_ramx.vhd"`, and `"mc8051_rom.vhd"` need to be copied from the directory `"../mc8051_boot/generate"` into the new directory as well. Since the Oregano IP Core was developed for a different hardware setup, a new PLL will need to be created to match the timing requirements of an 8051.

Once all the necessary files are in their respective folders, open the New Project Wizard window from within the Quartus Prime software and select the newly created directory as the project directory. Enter the exact name `"mc8051_top"` in the entity name field as seen in Figure 2.1

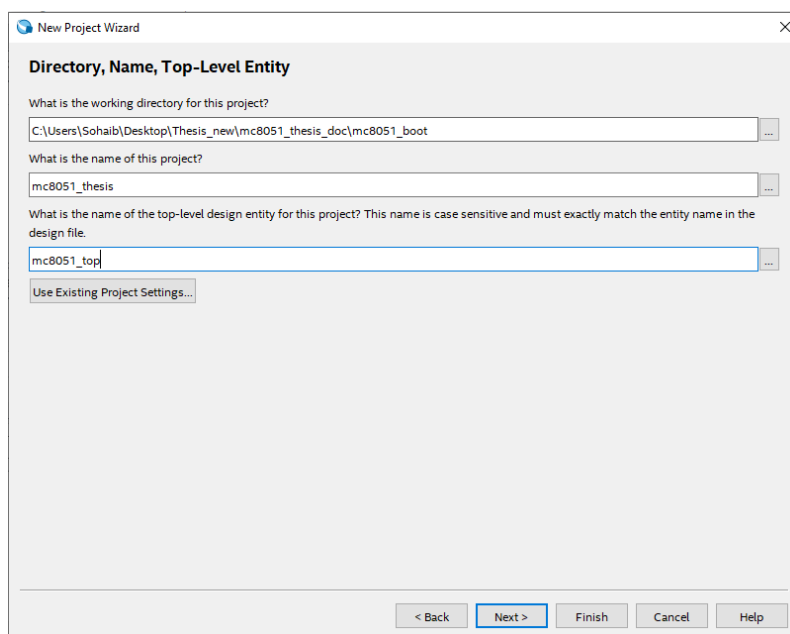


Figure 2.1: Quartus New Project Wizard, Directory selection

Now the file contents that were previously copied need to be added to the project. This can be done by clicking the "." button next to the filename filed in the Quartus New Project Wizard (Figure 2.3). All files previously copied now need to be added to the project.

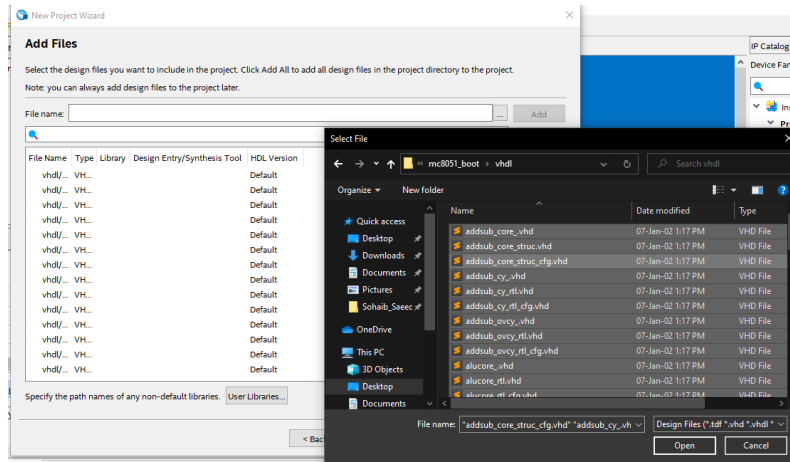


Figure 2.2: Quartus New Project Wizard, Adding Project files

After this is done, click "Next" to move onto the next step of the New Project Wizard.

In the *"Family, Device and Board Settings"* window, select *"Cyclone 10 LP"* as the device family. This provides a list of available devices to choose from. If the *"Cyclone 10 LP"* devices does not appear in the drop-down menu, the device package library for *"Cyclone 10 LP device support"* must be downloaded from the Quartus website [5].

Since the CYC1000 development board uses a 10CL025 FPGA, to easily find the correct board from the list, in the "Name" filter for the list of 10CL025 devices enter following name: *"10CL025YU256C8G"*. Select the device shown in the list and click "Next" to move onto the next step.

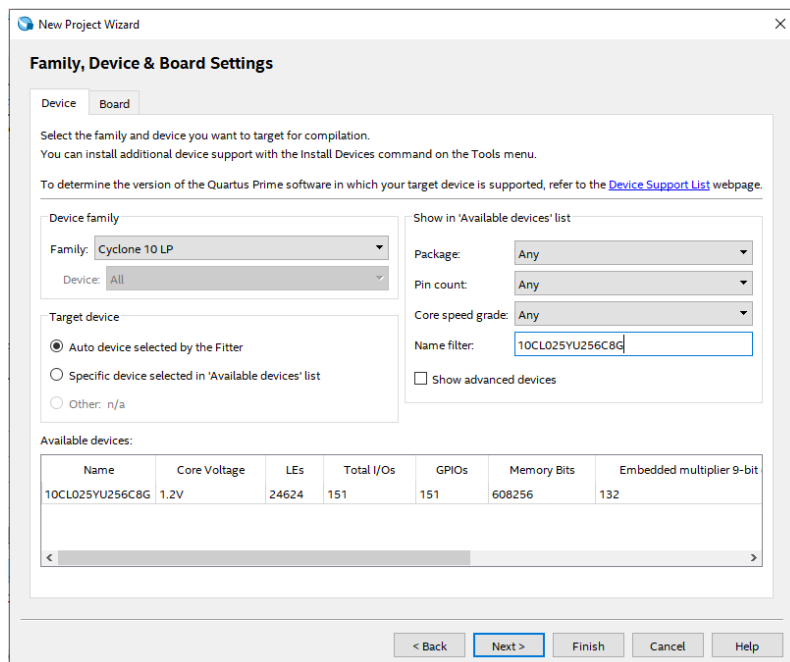


Figure 2.3: Quartus New Project Wizard, Selecting the CYC1000 development board

## 2.2 Creating the Cyclone PLL

The original design of the Oregano Systems 8051 IP Core uses a PLL to create an output clock frequency of 25 MHz. As we are using a newer version of quartus prime, the original PLL ".qip" file needs to be modified. To make things easier to understand, follow and debug, it is recommended to simply create a new PLL ".qip" file.

Firstly, create a new folder named "pll" in the same directory as the Quartus project file. Then, from the "IP Catalog" window on the right side of the Quartus Prime IDE, search for the term "pll" and select the "ATPLL" option from the list. This should open a new dialog box. Select "VHDL" as the file type and select the newly created "pll" folder as the path directory.

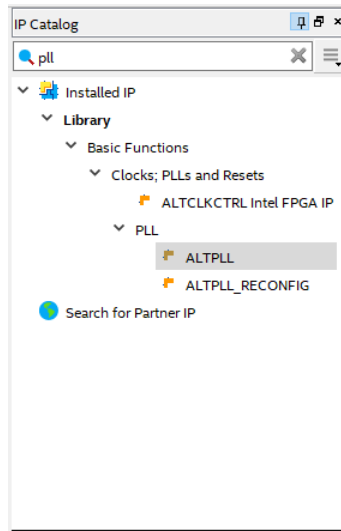


Figure 2.4: Creating the Cyclone PLL, ATPLL

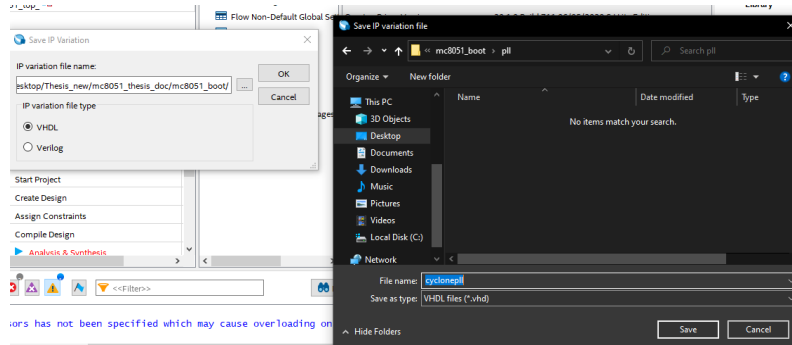


Figure 2.5: Creating the Cyclone PLL, selecting file directory

In the field "File name", write "*cyclonepll*" and click Save. Clicking "OK" on the dialog box will create the pll file and also open the "*MegaWizard*" window to help create the PLL.

Now the PLL must be configured according to our needs. In the "Input frequency" field (Page 1 of the Wizard), enter "12.000 MHz" as the PLL input clock and click "Next". This clock input would be later connected to the onboard 12 MHz clock of the CYC1000 development board.

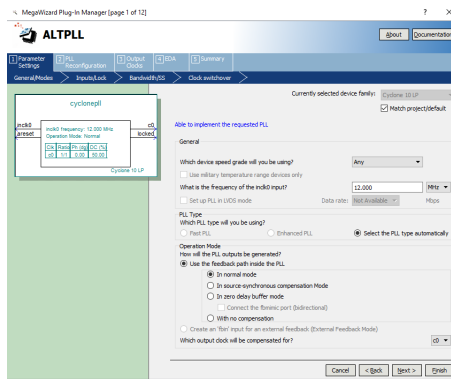


Figure 2.6: Creating the Cyclone PLL, Selecting the Input Frequency

In the next page (Page 2), deselect the *"Create an 'areset' input to asynchronously reset the PLL"* option and deselect the *"Create 'locked' output"* option and then click "Next" until you reach Page 6 of the Wizard.

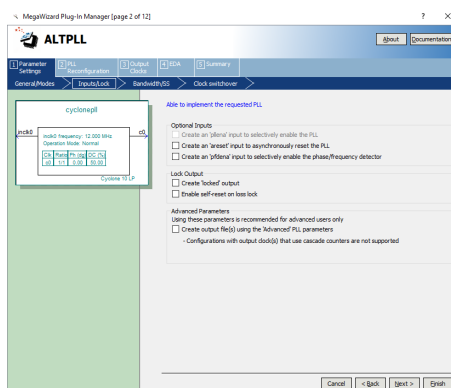


Figure 2.7: Creating the Cyclone PLL, Deselecting Asynchronous Reset

Now in the *c0-Core/External Output Clock* page of the Wizard (Page 6), select the option *"Enter output clock frequency"* and enter "25.000 MHz" as the desired output clock frequency. Click "Next" until you reach page 12 of the Wizard.

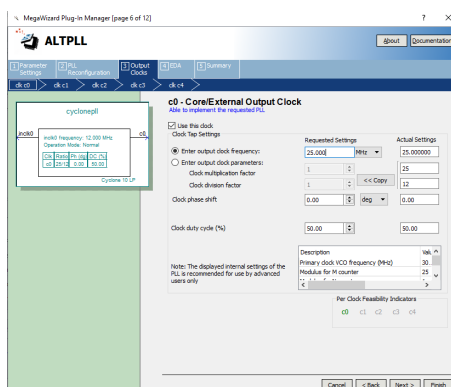


Figure 2.8: Creating the Cyclone PLL, Setting Output Clock Frequency



In the Summary page of the Wizard, deselect the "*cyclonepll.cmp*" option and click "Finish" to finish creating the PLL.

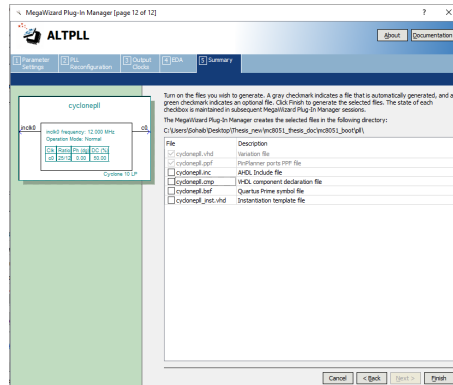


Figure 2.9: Creating the Cyclone PLL, Summary

The *cyclonepll.qip* file should already have been added to the project by default, but to be sure we can check by selecting "File" from the drop down menu in the Project Navigator window panel.

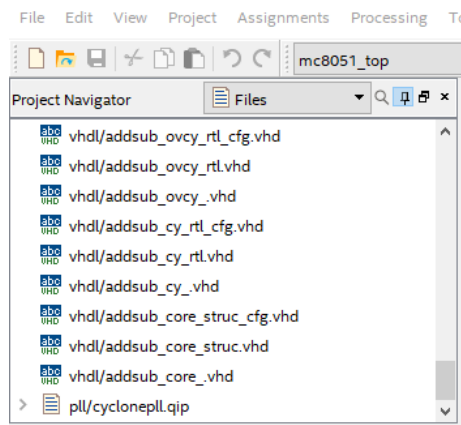


Figure 2.10: Creating the Cyclone PLL, Checking the *cyclonepll.qip* file

## 2.3 Necessary changes in the Top Level Architecture

When attempting to "Analyse and Synthesize", the output console shows some errors that are present in the *"mc8051\_top\_struct.vhd"* file (Figure 2.11). The reason for these errors is that the signal named *"s\_rom\_data"* is driven by two different modules, namely: *"mc8051\_datamux"* and *"mc8051\_rom"*. This can also be seen from the RTL view in the Quartus Prime software (Figure 2.12). To open the RTL view go to *Tools → Netlist Viewers → RTL Viewer*.

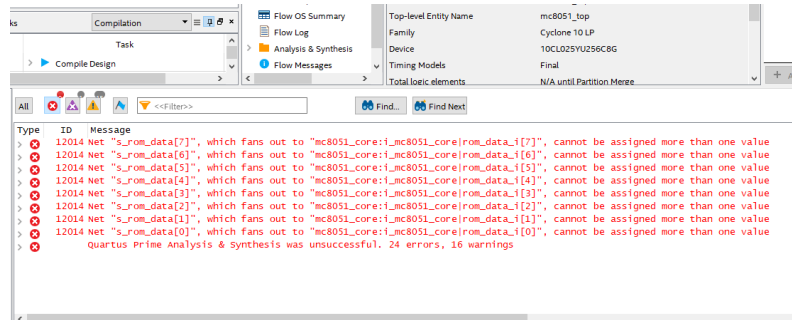


Figure 2.11: Oregano Systems 8051 IP Core Synthesis Errors

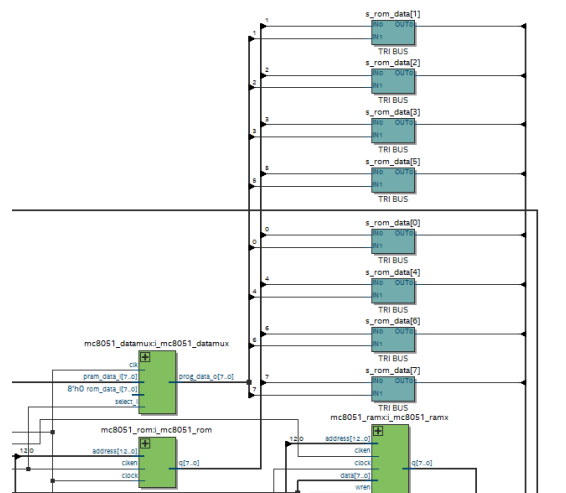


Figure 2.12: Oregano Systems 8051 IP Core RTL View

To solve this issue, in the file *"mc8051\_top\_struct.vhd"*, simply replace *"q => s\_rom\_data"* with *q => s\_pre\_rom\_data* (Line 154).

```

148 -----
149 -- Hook up the (up to) 64kx8 synchronous on-chip ROM.
150 i_mc8051_rom : mc8051_rom
151   port map (address => s_rom_adr_sml,
152             clock    => s_clk_pll,
153             clken    => s_rom_en,
154             q         => s_pre_rom_data);
155 -- THE ROM OF COURSE IS A MUST HAVE, ALTHOUGH THE SIZE CAN BE SMALLER!!
156 -----
157
158

```

Figure 2.13: Oregano Systems 8051 IP Core, replaced line 154

Once the line is replaced, the design should synthesise without any errors. This can also be verified in the RTL Viewer as well.

## 2.4 Pin Assignments

To compile and implement the final design in the CYC1000 development board, only the necessary I/O ports are assigned to fulfill the purpose of the thesis. For instance, to implement the FreeRTOS code and run different scheduling tasks in the FPGA, only the onboard LEDs are enough.

If necessary input ports such as  $P0\_i$  and so on can also be mapped to the FPGA.

To	Direction	Location	I/O Bank	VREF Group	I/O Standard
all_rxd_i[0]	Input	PIN_D15	6	B6_N0	3.3-V LVTTL
all_rxd_o[0]	Output	PIN_F15	6	B6_N0	3.3-V LVTTL
all_rxdwr_o[0]	Output				
all_t0_i[0]	Input	PIN_C15	6	B6_N0	3.3-V LVTTL
all_t1_i[0]	Input	PIN_B16	6	B6_N0	3.3-V LVTTL
all_txd_o[0]	Output				
altera_reserved_tck	Input				
altera_reserved_tdi	Input				
altera_reserved_tdo	Output				
altera_reserved_tms	Input				
clk	Input	PIN_M2	2	B2_N0	3.3-V LVTTL
int0_i[0]	Input	PIN_C16	6	B6_N0	3.3-V LVTTL
int1_i[0]	Input	PIN_D16	6	B6_N0	3.3-V LVTTL
p0_i[7]	Input				
p0_i[6]	Input				
p0_i[5]	Input				
p0_i[4]	Input				
p0_i[3]	Input				
p0_i[2]	Input				
p0_i[1]	Input				
p0_i[0]	Input				
p0_o[7]	Output				
p0_o[6]	Output				
p0_o[5]	Output				
p0_o[4]	Output				
p0_o[3]	Output				
p0_o[2]	Output				
p0_o[1]	Output				
p0_o[0]	Output				
p1_i[7]	Input				
p1_i[6]	Input				
p1_i[5]	Input				
p1_i[4]	Input				
p1_i[3]	Input				
p1_i[2]	Input				
p1_i[1]	Input				
p1_i[0]	Input				
p1_o[7]	Output				
p1_o[6]	Output				
p1_o[5]	Output				
p1_o[4]	Output				
p1_o[3]	Output				
p1_o[2]	Output				
p1_o[1]	Output				
p1_o[0]	Output				
p2_i[7]	Input				
p2_i[6]	Input				
p2_i[5]	Input				

Table 2.1 continued from previous page

To	Direction	Location	I/O Bank	VREF Group	I/O Standard
p2_i[4]	Input				
p2_i[3]	Input				
p2_i[2]	Input				
p2_i[1]	Input				
p2_i[0]	Input				
p2_o[7]	Output	PIN_N3	3	B3_N0	3.3-V LVTTL
p2_o[6]	Output	PIN_N5	3	B3_N0	3.3-V LVTTL
p2_o[5]	Output	PIN_R4	3	B3_N0	3.3-V LVTTL
p2_o[4]	Output	PIN_T2	3	B3_N0	3.3-V LVTTL
p2_o[3]	Output	PIN_R3	3	B3_N0	3.3-V LVTTL
p2_o[2]	Output	PIN_T3	3	B3_N0	3.3-V LVTTL
p2_o[1]	Output	PIN_T4	3	B3_N0	3.3-V LVTTL
p2_o[0]	Output	PIN_M6	3	B3_N0	3.3-V LVTTL
p3_i[7]	Input				
p3_i[6]	Input				
p3_i[5]	Input				
p3_i[4]	Input				
p3_i[3]	Input				
p3_i[2]	Input				
p3_i[1]	Input				
p3_i[0]	Input				
p3_o[7]	Output				
p3_o[6]	Output				
p3_o[5]	Output				
p3_o[4]	Output				
p3_o[3]	Output				
p3_o[2]	Output				
p3_o[1]	Output				
p3_o[0]	Output				
reset_n	Input	PIN_N6	3	B3_N0	3.3-V LVTTL

Table 2.1: Oregano Systems 8051 IP Core Pin Assignments

## 2.5 Design Compilation And Timing Analysis

When compiling, the design should compile without any errors and few warnings. However, there will be some critical warnings which occur because of inappropriate or undefined Timing constraints and uncertainties, as seen in Figure 2.14.

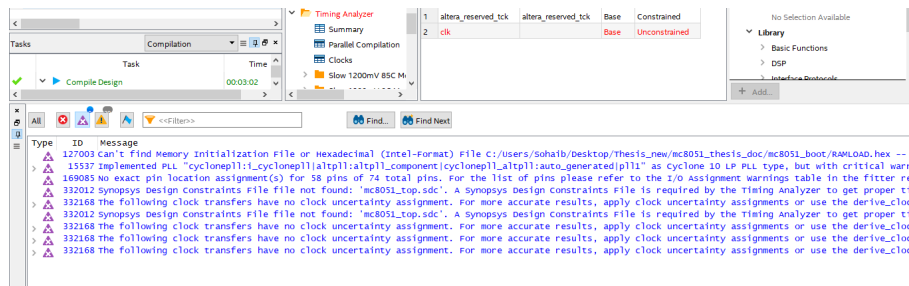


Figure 2.14: Timing Constraints - Critical Warnings

To overcome this issue, simply create a new empty text file with the contents as seen in Figure 2.15, and rename the extension to ".sdc"

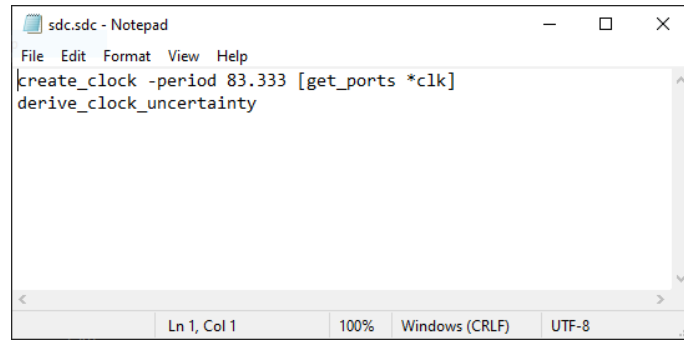


Figure 2.15: Creating the SDC file

This ".sdc" file must now be added to the project. This can be done by going to *Project* → *Add/Remove Files in Project*. Now attempting to synthesize the code will result in minor warnings that can easily be ignored for the purposes of this thesis.

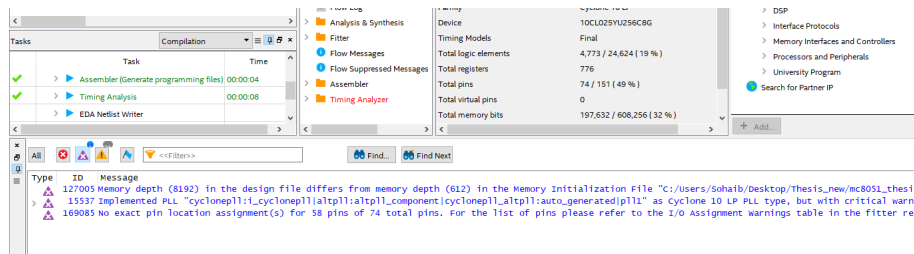


Figure 2.16: Synthesizing after implementing an SDC file

At this point, I decided that the next step should be to try to run the FreeRTOS 8051 Port - named the "Cygnal" port, on the Busy Bee 8051 Development Kit [9] to see what parts of the code need to be adapted to be able to run on the Oregano Systems 8051 IP Core. However, I ran into multiple errors when first compiling the Cygnal Port and as such, was forced to spend time debugging the port itself.

## Chapter 3

# Debugging the FreeRTOS Cygnal Port

### 3.1 Prerequisites

To be able to debug the Cygnal port, naturally the first step would be to download the Cygnal port. This can be done using the link mentioned in section 1.1: **List of Hardware and Software Requirements**.

Once the FreeRTOS source code has been downloaded, it can be extracted to a folder as such:

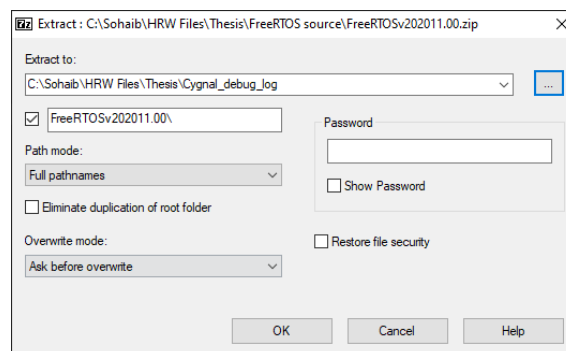


Figure 3.1: Extracting the FreeRTOS source code

The Cygnal Port is located under *"../FreeRTOSv202011.00/FreeRTOS/Demo/Cygnal"*.

Next, we need the GNU 'make' utility so that we can use the given Makefile to compile the source code. The download link is mentioned in section 1.1 as well. Once downloaded, ensure that the make utility is correctly placed in the same directory as the original Makefile (*"../FreeRTOS/Demo/Cygnal"*).

Name	Date modified	Type	Size
ParTest	11-Nov-20 9:24 PM	File folder	
serial	11-Nov-20 9:24 PM	File folder	
c8051f120.h	11-Nov-20 9:24 PM	H File	27 KB
FreeRTOSConfig.h	11-Nov-20 9:24 PM	H File	3 KB
main.c	11-Nov-20 9:24 PM	C File	16 KB
make.exe	05-Oct-03 4:49 PM	Application	165 KB
Makefile	11-Nov-20 9:24 PM	File	3 KB
sdcc.wsp	11-Nov-20 9:24 PM	WSP File	32 KB

Figure 3.2: GNU 'make' utility file placement

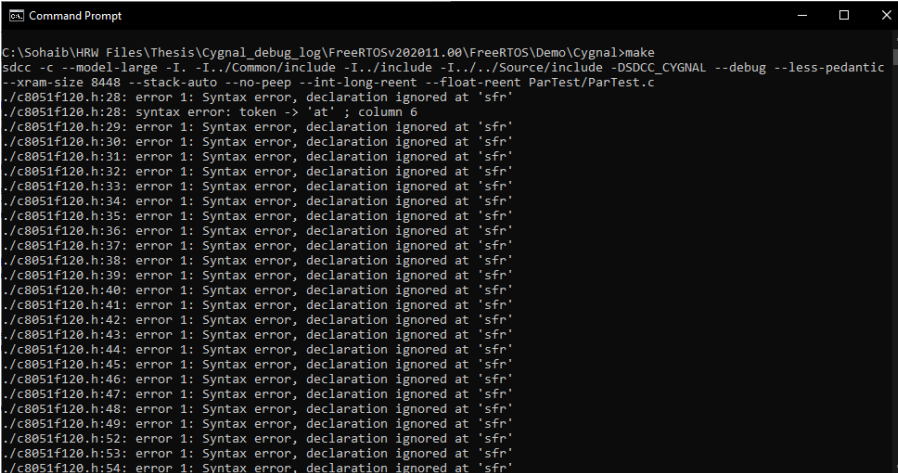
Lastly, SDCC version 4.0.0 or higher is required to compile the code. Once installed, typing `"sdcc -v"` in a command window will display the current installation status and versioning. Simply typing `"make"` in the directory of the Makefile and the GNU 'make' utility will start the compilation.

## 3.2 Changelog

This section of the report will feature every single change made to the FreeRTOS Cygnal port, it will attempt to do this in the most streamlined and easy to follow method possible. Featuring step by step instructions along with error code descriptions as well screenshots of said error codes. The error messages will be displayed as screenshots with the error description in the caption of the screenshot.

### 3.2.1 Changelog #1 : c8051f120.h

#### Issue



```

C:\Sohaib\HRW Files\Thesis\Cygnal_debug_log\FreeRTOSv202011.00\FreeRTOS\Demo\Cygnal>make
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ParTest/ParTest.c
./c8051f120.h:28: error 1: Syntax error, declaration ignored at 'sfr'
./c8051f120.h:28: syntax error: token -> 'at' ; column 6
./c8051f120.h:29: error 1: Syntax error, declaration ignored at 'sfr'
./c8051f120.h:30: error 1: Syntax error, declaration ignored at 'sfr'
./c8051f120.h:31: error 1: Syntax error, declaration ignored at 'sfr'
./c8051f120.h:32: error 1: Syntax error, declaration ignored at 'sfr'
./c8051f120.h:33: error 1: Syntax error, declaration ignored at 'sfr'
./c8051f120.h:34: error 1: Syntax error, declaration ignored at 'sfr'
./c8051f120.h:35: error 1: Syntax error, declaration ignored at 'sfr'
./c8051f120.h:36: error 1: Syntax error, declaration ignored at 'sfr'
./c8051f120.h:37: error 1: Syntax error, declaration ignored at 'sfr'
./c8051f120.h:38: error 1: Syntax error, declaration ignored at 'sfr'
./c8051f120.h:39: error 1: Syntax error, declaration ignored at 'sfr'
./c8051f120.h:40: error 1: Syntax error, declaration ignored at 'sfr'
./c8051f120.h:41: error 1: Syntax error, declaration ignored at 'sfr'
./c8051f120.h:42: error 1: Syntax error, declaration ignored at 'sfr'
./c8051f120.h:43: error 1: Syntax error, declaration ignored at 'sfr'
./c8051f120.h:44: error 1: Syntax error, declaration ignored at 'sfr'
./c8051f120.h:45: error 1: Syntax error, declaration ignored at 'sfr'
./c8051f120.h:46: error 1: Syntax error, declaration ignored at 'sfr'
./c8051f120.h:47: error 1: Syntax error, declaration ignored at 'sfr'
./c8051f120.h:48: error 1: Syntax error, declaration ignored at 'sfr'
./c8051f120.h:49: error 1: Syntax error, declaration ignored at 'sfr'
./c8051f120.h:52: error 1: Syntax error, declaration ignored at 'sfr'
./c8051f120.h:53: error 1: Syntax error, declaration ignored at 'sfr'
./c8051f120.h:54: error 1: Syntax error, declaration ignored at 'sfr'

```

Figure 3.3: Syntax error, declaration ignored at 'sfr', Syntax error, declaration ignored at 'sbit'

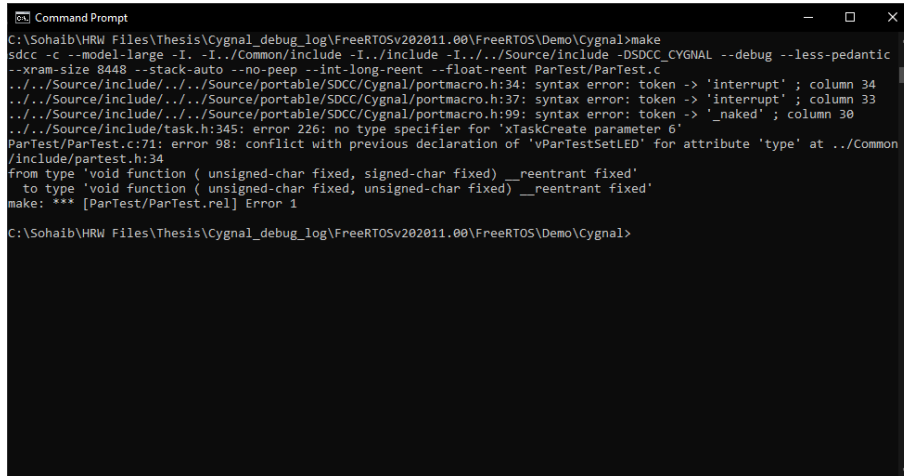
As seen in Figure 3.3, there already exists a slew of errors as the keyword `"sfr"` is not recognized by the compiler. This is an issue caused by the versioning of the compiler itself. The Cygnal Port was written for an SDCC version 2.4.0 or older and during my research I could not find the older variant of the SDCC. As such I was forced to use the 4.0.0 version and try to fix these issues with the current version of my compiler. It is my rough estimate that within the past 2 decades, the SDCC changed its keywords such that certain keywords such as `"sfr"` were no longer recognized and instead needed to be prepended with a double underscore (`"__"`). So the keyword `"sfr"` becomes `"__sfr"`. This needed to be done in multiple files for multiple keywords which are mentioned later on.

#### Fix

The fix itself can be implemented in any basic text editor with a "Find and Replace" feature. Simply open the file `"../FreeRTOS/Demo/Cygnal/c8051f120.h"` and replace all instances of the keyword `"sfr"` with `"__sfr"`, all instances of the keyword `"at"` with `"__at"` and all instances of the keyword `"sbit"` with `"__sbit"`.

### 3.2.2 Changelog #2 : portmacro.h

#### Issue #1



```
Command Prompt
C:\Sohaib\HRW Files\Thesis\Cygnal_debug_log\FreeRTOSv202011.00\FreeRTOS\Demo\Cygnal>make
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ParTest/ParTest.c
../Source/include/../../Source/portable/SDCC/Cygnal/portmacro.h:34: syntax error: token -> 'interrupt'; column 34
../Source/include/../../Source/portable/SDCC/Cygnal/portmacro.h:37: syntax error: token -> 'interrupt'; column 33
../Source/include/../../Source/portable/SDCC/Cygnal/portmacro.h:99: syntax error: token -> '_naked'; column 30
../Source/include/task.h:345: error 226: no type specifier for 'xTaskCreate' parameter 6'
ParTest/ParTest.c:71: error 98: conflict with previous declaration of 'vParTestSetLED' for attribute 'type' at ../Common
/include/portest.h:34
from type 'void function ( unsigned-char fixed, signed-char fixed) __reentrant fixed'
to type 'void function ( unsigned-char fixed, unsigned-char fixed) __reentrant fixed'
make: *** [ParTest/ParTest.rel] Error 1

C:\Sohaib\HRW Files\Thesis\Cygnal_debug_log\FreeRTOSv202011.00\FreeRTOS\Demo\Cygnal>
```

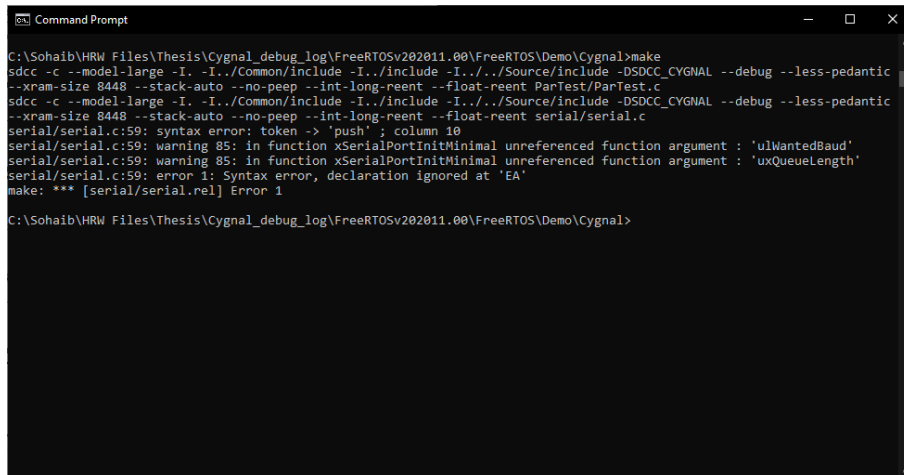
Figure 3.4: syntax error: token -> 'interrupt'

The keywords *"interrupt"* and *"naked"* are no longer supported by any version of SDCC 2.4.0 and higher.

#### Fix #1

Simply open the file *"../Source/portable/SDCC/portmacro.h"* and change the keywords *"interrupt"* and *"naked"* to *"\_\_interrupt"* and *"\_\_naked"* respectively.

#### Issue #2



```
Command Prompt
C:\Sohaib\HRW Files\Thesis\Cygnal_debug_log\FreeRTOSv202011.00\FreeRTOS\Demo\Cygnal>make
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ParTest/ParTest.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent serial/serial.c
serial/serial.c:59: syntax error: token -> 'push'; column 10
serial/serial.c:59: warning 85: in function xSerialPortInitMinimal unreferenced function argument : 'ulWantedBaud'
serial/serial.c:59: warning 85: in function xSerialPortInitMinimal unreferenced function argument : 'uxQueueLength'
serial/serial.c:59: error 1: Syntax error, declaration ignored at 'EA'
make: *** [serial/serial.rel] Error 1

C:\Sohaib\HRW Files\Thesis\Cygnal_debug_log\FreeRTOSv202011.00\FreeRTOS\Demo\Cygnal>
```

Figure 3.5: syntax error: token -> 'push' ;

Later on, after further changes were made to the code, an error in the file *"../FreeRTOS/Demo/-Cygnal/serial/serial.c"* related to a certain "push" token (line 59) needed to be resolved. The idea was to temporarily disable global interrupts so that a serial port could be successfully opened, and once done, re-enable the global interrupts. However macros defining interrupt controls are defined in file *"../Source/portable/SDCC/portmacro.h"* where the error actually occurs. As such, changes need to be made once again in *"portmacro.h"* to fix this issue.



The compiler fails due to a syntax error while defining the macros to disable or enable global interrupts. This is caused by the keywords `"_asm"` and `"_endasm"` no longer being supported by SDCC versions 2.4.0 or higher.

## Fix #2

Simply open the file `"../Source/portable/SDCC/portmacro.h"` and change the keywords `"_asm"` and `"_endasm"` to `"__asm"` and `"__endasm"` respectively.

## 3.2.3 Changelog #3 : task.h

### Issue

```

C:\Sohaib\HRW Files\Thesis\Cygnal_debug_log\FreeRTOSv202011.00\FreeRTOS\Demo\Cygnal>make
sdcc -c -model-large -I: ..../Common/include -I: ../include -I: ../Source/include -DSOCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ParTest/ParTest.c
../Source/include/task.h:345: error 226: no type specifier for 'xTaskCreate parameter 6'
ParTest/ParTest.c:71: error 98: conflict with previous declaration of 'vParTestSetLED' for attribute 'type' at ../Common
/include/parTest.h:34
from type 'void function ( unsigned-char fixed, signed-char fixed) __reentrant fixed'
to type 'void function ( unsigned-char fixed, unsigned-char fixed) __reentrant fixed'
make: *** [ParTest/ParTest.rel] Error 1

C:\Sohaib\HRW Files\Thesis\Cygnal_debug_log\FreeRTOSv202011.00\FreeRTOS\Demo\Cygnal>

```

Figure 3.6: error 226: no type specifier for 'xTaskCreate parameter 6'

The error here is caused by ~~(—improve—)~~ ... .

## Fix

Removing the `"const"` keyword in line 345 of file `"../FreeRTOS/Source/include/task.h"` removes the error.

```

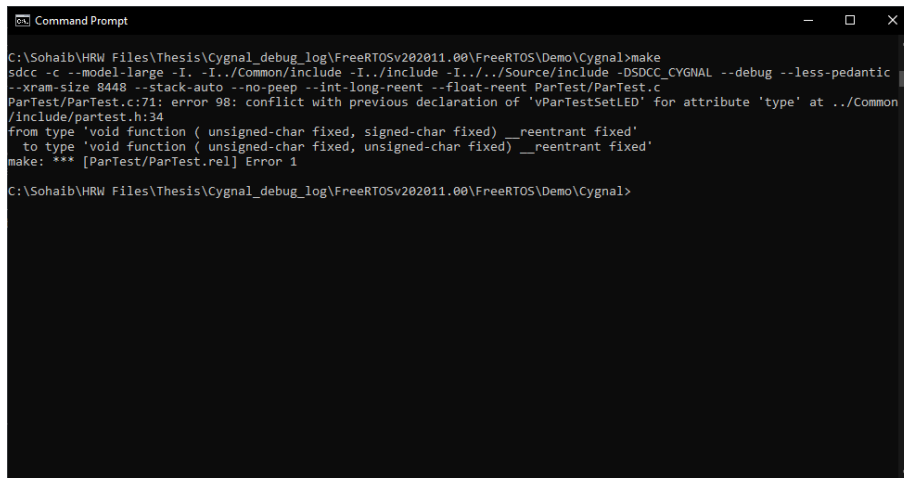
339 #if ( configSUPPORT_DYNAMIC_ALLOCATION == 1 )
340 BaseType_t xTaskCreate( TaskFunction_t pxTaskCode,
341     const char * const pcName, /*lint !e971 Unqualified char types are allowed for strings and single
342     const configSTACK_DEPTH_TYPE usStackDepth,
343     void * const pvParameters,
344     UBaseType_t uxPriority,
345     TaskHandle_t * pxCreatedTask ) PRIVILEGED_FUNCTION;
346 #endif

```

Figure 3.7: `"const"` keyword removed from line 345 (task.h)

### 3.2.4 Changelog #4 : ParTest.c

#### Issue #1



```
C:\Sohaib\HRW_Files\Thesis\Cygnal_debug_log\FreeRTOSv202011.00\FreeRTOS\Demo\Cygnal>make
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ParTest/ParTest.c
ParTest/ParTest.c:71: error 98: conflict with previous declaration of 'vParTestSetLED' for attribute 'type' at ../Common
/include/partest.h:34
from type 'void function ( unsigned-char fixed, signed-char fixed) __reentrant fixed'
to type 'void function ( unsigned-char fixed, unsigned-char fixed) __reentrant fixed'
make: *** [ParTest/ParTest.rel] Error 1

C:\Sohaib\HRW_Files\Thesis\Cygnal_debug_log\FreeRTOSv202011.00\FreeRTOS\Demo\Cygnal>
```

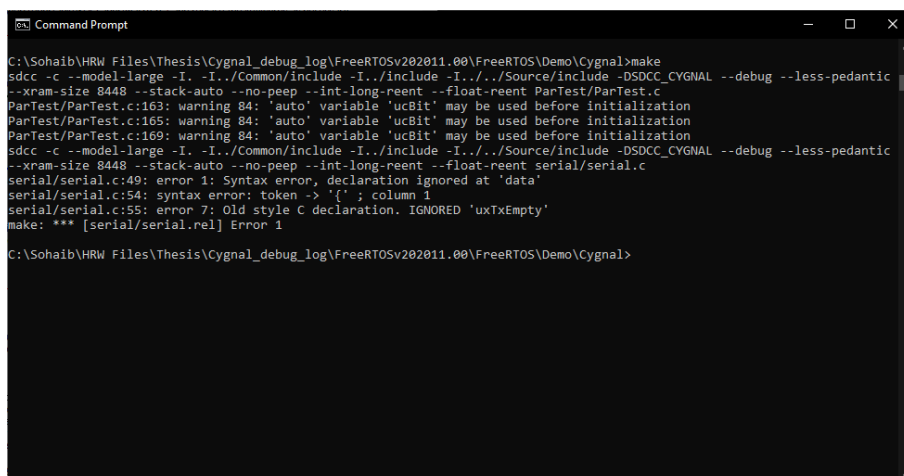
Figure 3.8: error 98: conflict with previous declaration of 'vParTestSetLED' for attribute 'type'

The error here is caused by a conflicting data type in the parameters of a FreeRTOS function. Normally the declaration of the function and the prototype of the function should contain the exact same data types. However, this is not the case here and there exists a discrepancy between the data types of the second parameter being passed to the function. The header file prototype sets the data type as *"signed-char fixed"* in file *"../FreeRTOS/Demo/Common/include/partest.h"*. However, *"unsigned-char fixed"* is used in the function declaration in file *"../FreeRTOS/Demo/Cygnal/ParTest/ParTest.c"* (Line 71).

#### Fix #1

The issue can be fixed by adding the keyword *"signed"* to the second parameter of the C function *"vParTestSetLED"* in file *"../FreeRTOS/Demo/Cygnal/ParTest/ParTest.c"*.

#### Issue #2



```
C:\Sohaib\HRW_Files\Thesis\Cygnal_debug_log\FreeRTOSv202011.00\FreeRTOS\Demo\Cygnal>make
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ParTest/ParTest.c
ParTest/ParTest.c:163: warning 84: 'auto' variable 'ucBit' may be used before initialization
ParTest/ParTest.c:169: warning 84: 'auto' variable 'ucBit' may be used before initialization
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent serial/serial.c
serial/serial.c:49: error 1: Syntax error, declaration ignored at 'data'
serial/serial.c:54: syntax error: token -> '(' ; column 1
serial/serial.c:55: error 7: Old style C declaration. IGNORED 'uxTxEmpty'
make: *** [serial/serial.rel] Error 1

C:\Sohaib\HRW_Files\Thesis\Cygnal_debug_log\FreeRTOSv202011.00\FreeRTOS\Demo\Cygnal>
```

Figure 3.9: warning 84: 'auto' variable 'ucBit' may be used before initialization

Additionally, a warning is generated by the SDCC as a variable declared within the function *"vParTestToggleLED"* named *"ucBit"* may be called or used before it can be initialized.

## Fix #2

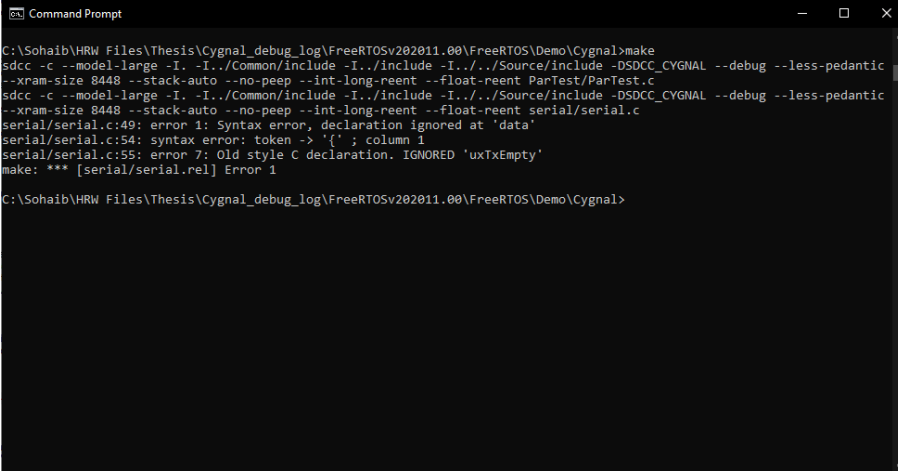
To remove the warning, simply comment out the declaration of this variable from within the scope of this function (line 133) and declare *"unsigned char ucBit"* as a global variable within file *"../FreeRTOS/Demo/Cygnal/ParTest/ParTest.c"*. (e.g. Line 46)

```
35
36 /* LED to output is dependent on how the LED's are wired. */
37 #define partstOUTPUT_0      ( ( unsigned char ) 0x02 )
38 #define partstOUTPUT_1      ( ( unsigned char ) 0x08 )
39 #define partstOUTPUT_2      ( ( unsigned char ) 0x20 )
40 #define partstOUTPUT_3      ( ( unsigned char ) 0x01 )
41 #define partstOUTPUT_4      ( ( unsigned char ) 0x04 )
42 #define partstOUTPUT_5      ( ( unsigned char ) 0x10 )
43 #define partstOUTPUT_6      ( ( unsigned char ) 0x40 )
44 #define partstOUTPUT_7      ( ( unsigned char ) 0x80 )
45
46 unsigned char ucBit; // -- Added to remove warning (Changelog #4)
47
48 /*-----
49  * Simple parallel port IO routines.
50  *-----*/
```

Figure 3.10: *"unsigned char ucBit"* added to line 46 (ParTest.c)

## 3.2.5 Changelog #5 : serial.c

### Issue #1



```
Command Prompt
C:\Sohaib\HRW Files\Thesis\Cygnal_debug_log\FreeRTOSv202011.00\FreeRTOS\Demo\Cygnal>make
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ParTest/ParTest.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent serial/serial.c
serial/serial.c:49: error 1: Syntax error, declaration ignored at 'data'
serial/serial.c:54: syntax error: token -> '{' ; column 1
serial/serial.c:55: error 7: Old style C declaration. IGNORED 'uxTxEmpty'
make: *** [serial/serial.rel] Error 1

C:\Sohaib\HRW Files\Thesis\Cygnal_debug_log\FreeRTOSv202011.00\FreeRTOS\Demo\Cygnal>
```

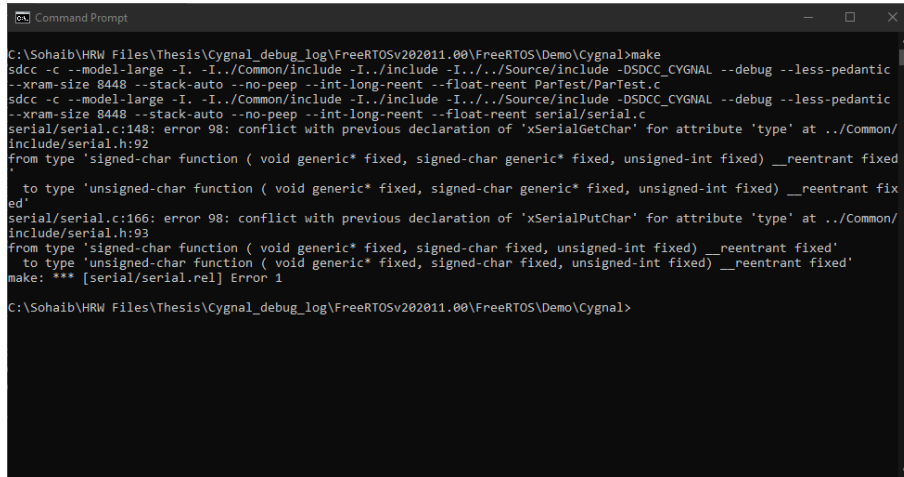
Figure 3.11: Syntax error, declaration ignored at 'data'

The keywords *"data"* and *"interrupt"* are no longer supported by any version of SDCC 2.4.0 and higher.

## Fix #1

Simply open the file *"../FreeRTOS/Demo/Cygnal/serial/serial.c"* and change the keywords *"data"* and *"interrupt"* to *"\_\_data"* and *"\_\_interrupt"* respectively (Line 49, and line 102).

## Issue #2



```
C:\Sohaib\HRW Files\Thesis\Cygnal_debug_log\FreeRTOSv202011.00\FreeRTOS\Demo\Cygnal>make
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ParTest/ParTest.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent serial/serial.c
serial/serial.c:148: error 98: conflict with previous declaration of 'xSerialGetChar' for attribute 'type' at ../Common/
include/serial.h:92
from type 'signed-char function ( void generic* fixed, signed-char generic* fixed, unsigned-int fixed) __reentrant fixed'
to type 'unsigned-char function ( void generic* fixed, signed-char generic* fixed, unsigned-int fixed) __reentrant fixed'
serial/serial.c:166: error 98: conflict with previous declaration of 'xSerialPutChar' for attribute 'type' at ../Common/
include/serial.h:93
from type 'signed-char function ( void generic* fixed, signed-char fixed, unsigned-int fixed) __reentrant fixed'
to type 'unsigned-char function ( void generic* fixed, signed-char fixed, unsigned-int fixed) __reentrant fixed'
make: *** [serial/serial.rel] Error 1

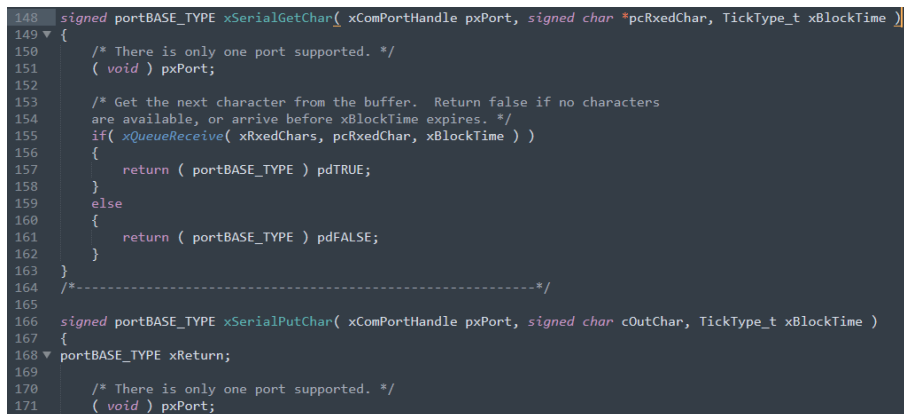
C:\Sohaib\HRW Files\Thesis\Cygnal_debug_log\FreeRTOSv202011.00\FreeRTOS\Demo\Cygnal>
```

Figure 3.12: error 98: conflict with previous declaration of 'xSerialGetChar' for attribute 'type'

This error is caused by a discrepancy between the return types of the functions *"xSerialGetChar"* and *"xSerialPutChar"* as defined in the header file *"../FreeRTOS/Demo/Common/include/serial.h"* and the C source file *"../FreeRTOS/Demo/Cygnal/serial/serial.c"*

## Fix #2

Simply open the file *"../FreeRTOS/Demo/Cygnal/serial/serial.c"* and add the keyword *"signed"* as the return data type for both the *"xSerialGetChar"* and *"xSerialPutChar"* functions (Line 148 and 166).

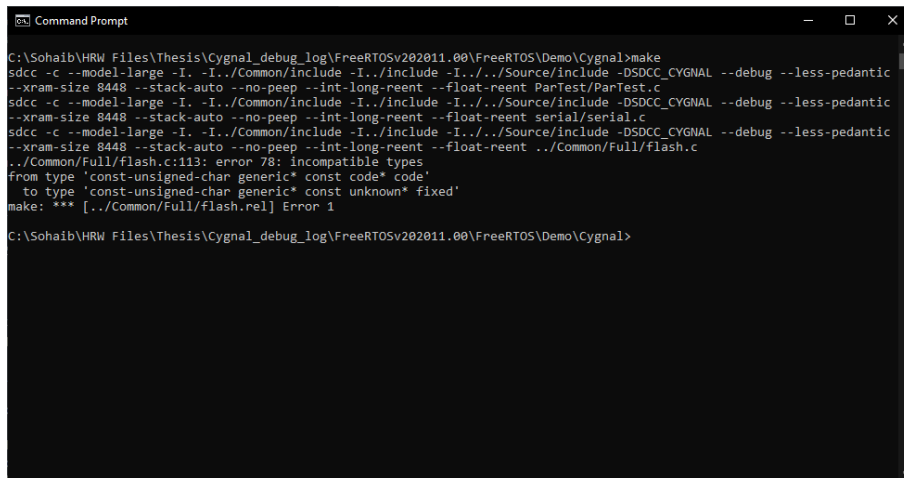


```
148 signed portBASE_TYPE xSerialGetChar( xComPortHandle pxPort, signed char *pcRxedChar, TickType_t xBlockTime )
149 {
150     /* There is only one port supported. */
151     ( void ) pxPort;
152
153     /* Get the next character from the buffer. Return false if no characters
154     are available, or arrive before xBlockTime expires. */
155     if( xQueueReceive( xRxedChars, pcRxedChar, xBlockTime ) )
156     {
157         return ( portBASE_TYPE ) pdTRUE;
158     }
159     else
160     {
161         return ( portBASE_TYPE ) pdFALSE;
162     }
163 }
164 /*-----*/
165
166 signed portBASE_TYPE xSerialPutChar( xComPortHandle pxPort, signed char cOutChar, TickType_t xBlockTime )
167 {
168     portBASE_TYPE xReturn;
169
170     /* There is only one port supported. */
171     ( void ) pxPort;
```

Figure 3.13: *"signed"* keyword added to line 148 and 166 (serial.c)

### 3.2.6 Changelog #6 : flash.c

#### Issue



```
C:\Sohaib\HRW Files\Thesis\Cygnal_debug_log\FreeRTOSv202011.00\FreeRTOS\Demo\Cygnal>make
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ParTest/ParTest.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent serial/serial.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Full/flash.c
../Common/Full/flash.c:113: error 78: incompatible types
from type 'const-unsigned-char generic* const code* code'
to type 'const-unsigned-char generic* const unknown* fixed'
make: *** [../Common/Full/flash.rel] Error 1

C:\Sohaib\HRW Files\Thesis\Cygnal_debug_log\FreeRTOSv202011.00\FreeRTOS\Demo\Cygnal>
```

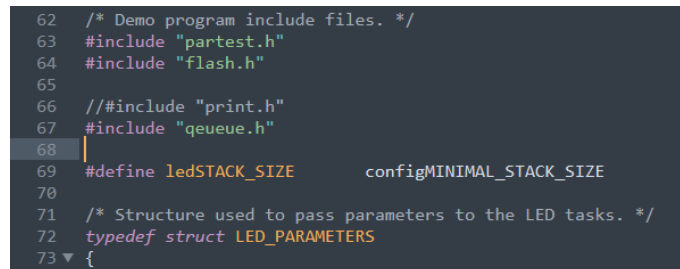
Figure 3.14: error 78: incompatible types

The error here is caused by a known bug in the SDCC compiler [12]. An error occurs when including the `"../FreeRTOS/Demo/Common/include/print.h"` file and the function `"vPrintDisplayMessage"` cannot be successfully implemented as it causes some type of type conflict.

#### Fix

To fix this error, I needed to copy the function defined in the `"print.c"` file and paste it as a local function within the `"flash.c"` file instead of importing the function from `"print.h"`. Additionally, as the function from `"print.c"` makes use of the `"queue.h"` header file, I needed to include this as well within `"flash.c"`.

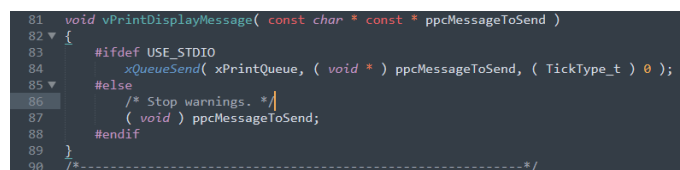
In file `"../FreeRTOS/Demo/Common/Full/flash.c"`, remove the inclusion of `"print.h"` and include `"queue.h"`.



```
62 /* Demo program include files. */
63 #include "partest.h"
64 #include "flash.h"
65
66 //#include "print.h"
67 #include "queue.h"
68
69 #define ledSTACK_SIZE configMINIMAL_STACK_SIZE
70
71 /* Structure used to pass parameters to the LED tasks. */
72 typedef struct LED_PARAMETERS
73 {
```

Figure 3.15: commented out include for `"print.h"` and included `"queue.h"` (flash.c)

After this has been done, from file `"../FreeRTOS/Demo/Common/Full/print.c"`, copy the `"vPrintDisplayMessage"` function (Line 81 - 89).



```
81 void vPrintDisplayMessage( const char * const * ppcMessageToSend )
82 {
83 /* #ifdef USE_STDIO
84 xQueueSend( xPrintQueue, ( void * ) ppcMessageToSend, ( TickType_t ) 0 );
85 /* #else
86 /* Stop warnings. */
87 /* ( void ) ppcMessageToSend;
88 /* #endif
89 }
90 /* -----*/
```

Figure 3.16: copy function `"vPrintDisplayMessage"` (print.c)

Once the function has been copied, it needs to be commented out from file `"../FreeRTOS/Demo/Common/Full/print.c"` as well as the header file `"../FreeRTOS/Demo/Common/include/print.h"`

(Line 32).

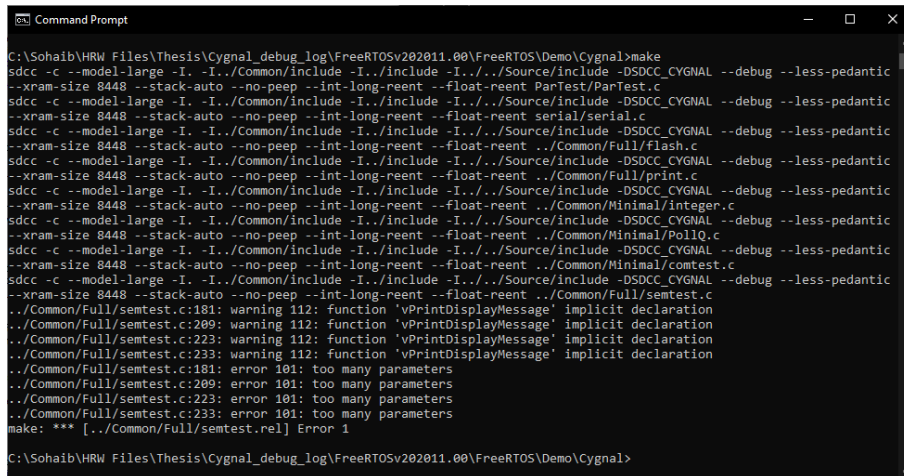
Now place the original copied function within `"../FreeRTOS/Demo/Common/Full/flash.c"` above the function `"vLEDFlashTask"` and rename the function to `"my_vPrintDisplayMessage"`. This is done to fix a later error which cannot allow multiple local function definitions with the same name, even if they are not being included or referenced in any way. As we have renamed the function, the function call at line 125 (in function `"vLEDFlashTask"`) must also then be changed accordingly.

```
110 void my_vPrintDisplayMessage( const char * const * ppcMessageToSend )
111 {
112     #ifdef USE_STDIO
113         xQueueSend( xPrintQueue, ( void * ) ppcMessageToSend, ( TickType_t ) 0 );
114     #else
115         /* Stop warnings. */
116         ( void ) ppcMessageToSend;
117     #endif
118 }
119
120 static void vLEDFlashTask( void *pvParameters )
121 {
122     xLEDPParameters *pxParameters;
123
124     /* Queue a message for printing to say the task has started. */
125     my_vPrintDisplayMessage( 8pcTaskStartMsg );
126
127     pxParameters = ( xLEDPParameters * ) pvParameters;
128
129     for(;;)
130     {
```

Figure 3.17: paste function `"vPrintDisplayMessage"` and rename to `"my_vPrintDisplayMessage"`, and edit function call (flash.c)

### 3.2.7 Changelog #7 : semtest.c

#### Issue



```
C:\Sohaib\HRW Files\Thesis\Cygnal_debug_log\FreeRTOSv202011.00\FreeRTOS\Demo\Cygnal>make
sdcc -c --model-large -I. -I../Common/include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ParTest/ParTest.c
sdcc -c --model-large -I. -I../Common/include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent serial/serial.c
sdcc -c --model-large -I. -I../Common/include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Full/flash.c
sdcc -c --model-large -I. -I../Common/include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Full/print.c
sdcc -c --model-large -I. -I../Common/include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Minimal/integer.c
sdcc -c --model-large -I. -I../Common/include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Minimal/PollQ.c
sdcc -c --model-large -I. -I../Common/include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Minimal/comtest.c
sdcc -c --model-large -I. -I../Common/include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Full/semtest.c
../Common/Full/semtest.c:181: warning 112: function 'vPrintDisplayMessage' implicit declaration
../Common/Full/semtest.c:209: warning 112: function 'vPrintDisplayMessage' implicit declaration
../Common/Full/semtest.c:223: warning 112: function 'vPrintDisplayMessage' implicit declaration
../Common/Full/semtest.c:233: warning 112: function 'vPrintDisplayMessage' implicit declaration
../Common/Full/semtest.c:181: error 101: too many parameters
../Common/Full/semtest.c:209: error 101: too many parameters
../Common/Full/semtest.c:223: error 101: too many parameters
../Common/Full/semtest.c:233: error 101: too many parameters
make: *** [../Common/Full/semtest.rel] Error 1

C:\Sohaib\HRW Files\Thesis\Cygnal_debug_log\FreeRTOSv202011.00\FreeRTOS\Demo\Cygnal>
```

Figure 3.18: warning 112: function 'vPrintDisplayMessage' implicit declaration, error 101: too many parameters

The error here is caused by the usage of the function `"vPrintDisplayMessage"` where no such function exists, as it has been commented out in the previous changelog.

## Fix

To fix this issue, firstly within the file `"../FreeRTOS/Demo/Common/Full/semtest.c"`, remove the inclusion of `"print.h"` and include `"queue.h"`.

```
81 /* Demo app include files. */
82 #include "semtest.h"
83 // #include "print.h"
84
85 #include "queue.h"
86
87 /* The value to which the shared variables are counted. */
88 #define semtstBLOCKING_EXPECTED_VALUE ( ( unsigned long ) 0xff )
89 #define semtstNON_BLOCKING_EXPECTED_VALUE ( ( unsigned long ) 0xff )
```

Figure 3.19: commented out include for `"print.h"` and included `"queue.h"` (semtest.c)

Now simply copy the same `"vPrintDisplayMessage"` function, as seen in Figure 3.16, from file `"../FreeRTOS/Demo/Common/Full/print.c"` (Line 81 - 89), and paste in file `"../FreeRTOS/Demo/Common/Full/semtest.c"`, above the function `"prvSemaphoreTest"` (Line 168)

```
167
168 void vPrintDisplayMessage( const char * const * ppcMessageToSend )
169 {
170     #ifdef USE_STDIO
171         xQueueSend( xPrintQueue, ( void * ) ppcMessageToSend, ( TickType_t ) 0 );
172     #else
173         /* Stop warnings. */
174         ( void ) ppcMessageToSend;
175     #endif
176 }
177
178 static void prvSemaphoreTest( void *pvParameters )
179 {
180     xSemaphoreParameters *pxParameters;
181     volatile unsigned long *pulSharedVariable, ulExpectedValue;
182     unsigned long ulCounter;
183     short sError = pdFALSE, sCheckVariableToUse;
```

Figure 3.20: paste function `"vPrintDisplayMessage"` as is, without changing anything else (semtest.c)

## 3.2.8 Changelog #8 : tasks.c

### Issue

```
Command Prompt
C:\Sohaib\HRW\Files\Cygnal_debug_log\FreeRTOSv202011.00\FreeRTOS\Demo\Cygnal>make
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ParTest/ParTest.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent serial/serial.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Full/flash.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Full/print.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Minimal/integer.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Minimal/PollQ.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Minimal/comtest.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Full/semtest.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Source/tasks.c
../Source/tasks.c:550: error 226: no type specifier for 'prvInitialiseNewTask parameter 6'
../Source/tasks.c:732: error 226: no type specifier for 'xTaskCreate parameter 6'
../Source/tasks.c:726: error 98: conflict with previous declaration of 'xTaskCreate' for attribute 'type' at ../Source/tasks.c:340
from type 'signed-char function ( void function ( void generic* fixed ) __reentrant code* fixed, const-unsigned-char gene
ric* const fixed, const-unsigned-int fixed, void generic* const fixed, unsigned-char fixed, struct tskTaskControlBlock g
eneric* generic* fixed ) __reentrant fixed'
to type 'signed-char function ( void function ( void generic* fixed ) __reentrant code* fixed, const-unsigned-char gene
ric* const fixed, const-unsigned-int fixed, void generic* const fixed, unsigned-char fixed, struct tskTaskControlBlock g
eneric* int generic* const fixed ) __reentrant fixed'
../Source/tasks.c:825: error 226: no type specifier for 'prvInitialiseNewTask parameter 6'
```

Figure 3.21: error 226: no type specifier for `'prvInitialiseNewTask parameter 6'`

I do not know exactly what causes this error. However, through an online resource I was able to find a fix.

## Fix

Removing the *"const"* keyword in lines 548, 731 and 822 of file *"../FreeRTOS/Source/tasks.c"* fixes the error

```
539 ▾ /*  
540  * Called after a Task_t structure has been allocated either statically or  
541  * dynamically to fill in the structure's members.  
542  */  
543 ▾ static void prvInitialiseNewTask( TaskFunction_t pxTaskCode,  
544                                     const char * const pcName, /*lint !e971 Unqualified char types are allowed for strings  
545                                     const uint32_t ulStackDepth,  
546                                     void * const pvParameters,  
547                                     UBaseType_t uxPriority,  
548                                     TaskHandle_t * pxCreatedTask, // used to be - TaskHandle_t * const pxCreatedTask  
549                                     TCB_t * pxNewTCB,  
550                                     const MemoryRegion_t * const xRegions ) PRIVILEGED_FUNCTION;  
551  
552 ▾ /*  
553  * Called after a new task has been created and initialised to place the task
```

Figure 3.22: *"const"* keyword removed from line 548 (tasks.c)

```
724 #if ( configSUPPORT_DYNAMIC_ALLOCATION == 1 )  
725  
726 BaseType_t xTaskCreate( TaskFunction_t pxTaskCode,  
727                         const char * const pcName, /*lint !e971 Unqualified char types are allowed for strings  
728                         const configSTACK_DEPTH_TYPE usStackDepth,  
729                         void * const pvParameters,  
730                         UBaseType_t uxPriority,  
731                         TaskHandle_t * pxCreatedTask ) // used to be - TaskHandle_t * const pxCreatedTask  
732 {  
733     TCB_t * pxNewTCB;  
734     BaseType_t xReturn;  
735
```

Figure 3.23: *"const"* keyword removed from line 731 (tasks.c)

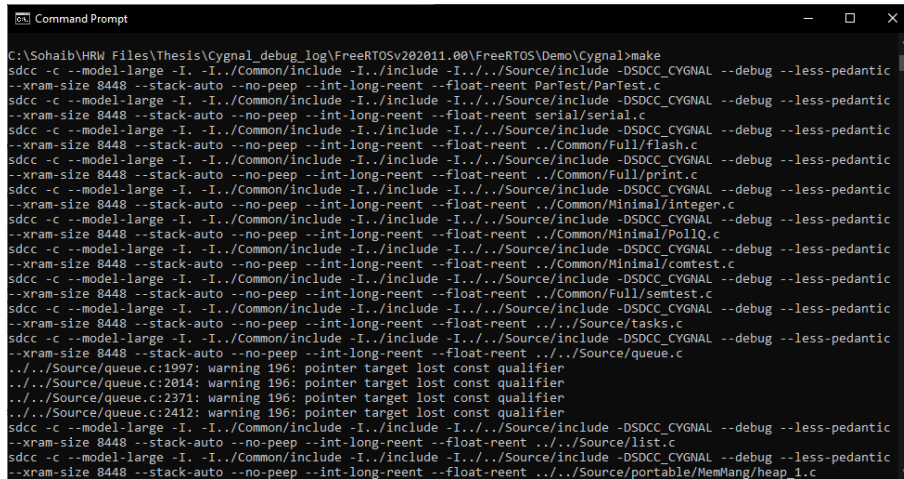
```
816  
817 ▾ static void prvInitialiseNewTask( TaskFunction_t pxTaskCode,  
818                                     const char * const pcName, /*lint !e971 Unqualified char types are allowed for strings  
819                                     const uint32_t ulStackDepth,  
820                                     void * const pvParameters,  
821                                     UBaseType_t uxPriority,  
822                                     TaskHandle_t * pxCreatedTask, // used to be - TaskHandle_t * const pxCreatedTask  
823                                     TCB_t * pxNewTCB,  
824                                     const MemoryRegion_t * const xRegions )  
825 {  
826     StackType_t * pxTopOfStack;  
827     UBaseType_t x;  
828
```

Figure 3.24: *"const"* keyword removed from line 822 (tasks.c)



### 3.2.9 Changelog #9 : queue.c

#### Issue



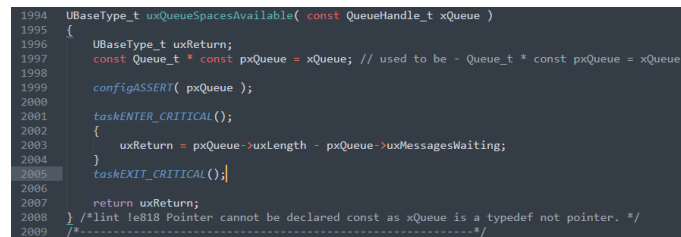
```
C:\Sohaib\HRW_Files\Thesis\Cygnal_debug_log\FreeRTOSv202011.00\FreeRTOS\Demo\Cygnal>make
sdcc -c -model-large -I -I./Common/include -I./include -I./../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ParTest/ParTest.c
sdcc -c -model-large -I -I./Common/include -I./include -I./../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent serial/serial.c
sdcc -c -model-large -I -I./Common/include -I./include -I./../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Full/flash.c
sdcc -c -model-large -I -I./Common/include -I./include -I./../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Full/print.c
sdcc -c -model-large -I -I./Common/include -I./include -I./../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Minimal/integer.c
sdcc -c -model-large -I -I./Common/include -I./include -I./../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Minimal/PollQ.c
sdcc -c -model-large -I -I./Common/include -I./include -I./../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Minimal/comtest.c
sdcc -c -model-large -I -I./Common/include -I./include -I./../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Full/semtest.c
sdcc -c -model-large -I -I./Common/include -I./include -I./../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Source/tasks.c
sdcc -c -model-large -I -I./Common/include -I./include -I./../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Source/queue.c
../Source/queue.c:1997: warning 196: pointer target lost const qualifier
../Source/queue.c:2014: warning 196: pointer target lost const qualifier
../Source/queue.c:2371: warning 196: pointer target lost const qualifier
../Source/queue.c:2412: warning 196: pointer target lost const qualifier
sdcc -c -model-large -I -I./Common/include -I./include -I./../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Source/list.c
sdcc -c -model-large -I -I./Common/include -I./include -I./../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Source/portable/MemMang/heap_1.c
```

Figure 3.25: warning 196: pointer target lost const qualifier

The warning here is caused by a mismatching of the "*const*" identifier of a struct. The error occurs when a function argument takes a "*const*" struct and assigns a non-constant pointer to this struct.

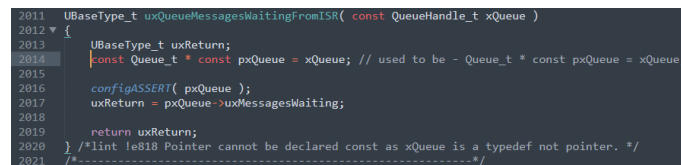
#### Fix

To fix this issue, lines 1997, 2014, 2371 and 2412 of file "*../FreeRTOS/Source/queue.c*" need to be changed such that the pointer assignment also contains the "*const*" identifier.



```
1994 UBaseType_t uxQueueSpacesAvailable( const QueueHandle_t xQueue )
1995 {
1996     UBaseType_t uxReturn;
1997     const Queue_t * const pxQueue = xQueue; // used to be - Queue_t * const pxQueue = xQueue;
1998
1999     configASSERT( pxQueue );
2000
2001     taskENTER_CRITICAL();
2002     {
2003         uxReturn = pxQueue->uxLength - pxQueue->uxMessagesWaiting;
2004     }
2005     taskEXIT_CRITICAL();
2006
2007     return uxReturn;
2008 } /*lint !e818 Pointer cannot be declared const as xQueue is a typedef not pointer. */
2009 /*-----*/
```

Figure 3.26: "*const*" keyword added to line 1997 (queue.c)



```
2011 UBaseType_t uxQueueMessagesWaitingFromISR( const QueueHandle_t xQueue )
2012 {
2013     UBaseType_t uxReturn;
2014     const Queue_t * const pxQueue = xQueue; // used to be - Queue_t * const pxQueue = xQueue;
2015
2016     configASSERT( pxQueue );
2017     uxReturn = pxQueue->uxMessagesWaiting;
2018
2019     return uxReturn;
2020 } /*lint !e818 Pointer cannot be declared const as xQueue is a typedef not pointer. */
2021 /*-----*/
```

Figure 3.27: "*const*" keyword added to line 2014 (queue.c)

```

2368 BaseType_t xQueueIsQueueEmptyFromISR( const QueueHandle_t xQueue )
2369 {
2370     BaseType_t xReturn;
2371     const Queue_t * const pxQueue = xQueue; // used to be - Queue_t * const pxQueue = xQueue;
2372
2373     configASSERT( pxQueue );
2374
2375     if( pxQueue->uxMessagesWaiting == ( UBaseType_t ) 0 )
2376     {
2377         xReturn = pdTRUE;
2378     }
2379     else
2380     {
2381         xReturn = pdFALSE;
2382     }
2383
2384     return xReturn;
2385 } /*lint !e818 xQueue could not be pointer to const because it is a typedef. */
2386 /*-----*/

```

Figure 3.28: "*const*" keyword added to line 2371 (queue.c)

```

2409 BaseType_t xQueueIsQueueFullFromISR( const QueueHandle_t xQueue )
2410 {
2411     BaseType_t xReturn;
2412     const Queue_t * const pxQueue = xQueue; // used to be - Queue_t * const pxQueue = xQueue;
2413
2414     configASSERT( pxQueue );
2415
2416     if( pxQueue->uxMessagesWaiting == pxQueue->uxLength )
2417     {
2418         xReturn = pdTRUE;
2419     }
2420     else
2421     {
2422         xReturn = pdFALSE;
2423     }
2424
2425     return xReturn;
2426 } /*lint !e818 xQueue could not be pointer to const because it is a typedef. */
2427 /*-----*/

```

Figure 3.29: "*const*" keyword added to line 2412 (queue.c)

### 3.2.10 Changelog #10 : heap\_1.c

#### Issue

```

Command Prompt
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent serial/serial.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Full/flash.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Full/print.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Minimal/integer.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Minimal/Poll.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Minimal/contest.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Full/semtest.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Source/tasks.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Source/queue.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Source/list.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Source/portable/MemMang/heap_1.c
../Source/portable/MemMang/heap_1.c:91: warning 127: non-pointer type cast to generic pointer
from type 'unsigned-long-int register'
to type 'unsigned-char generic* fixed'
../Source/portable/MemMang/heap_1.c:91: warning 127: non-pointer type cast to generic pointer
from type 'unsigned-long-int fixed'
to type 'unsigned-char generic* fixed'
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Source/portable/SDCC/Cygnal/port.c
../Source/portable/SDCC/Cygnal/port.c:56: error 1: Syntax error, declaration ignored at 'data'

```

Figure 3.30: warning 127: non-pointer type cast to generic pointer

The warning here is caused by an illegal pointer typecast operation in the function *"pvPortMalloc"* in file *"../FreeRTOS/Source/portable/MemMang/heap\_1.c"* (Line 91).

```

86     vTaskSuspendAll();
87     {
88         if( pucAlignedHeap == NULL )
89         {
90             /* Ensure the heap starts on a correctly aligned boundary. */
91             pucAlignedHeap = ( uint8_t * ) ( ( ( portPOINTER_SIZE_TYPE ) & ucHeap[ portBYTE_ALIGNMENT ] ) & ~( ( portPOINTER_SIZE
92             )
93

```

Figure 3.31: Illegal pointer typecast operation in line 91 (heap\_1.c)

## Fix

Unfortunately I was not able to find a fix for this particular warning. However the code seems to run fine without any further changes. Nevertheless, I felt it was necessary to mention it here.

### 3.2.11 Changelog #11 : port.c

#### Issue

```

C:\Sohaib\HRW Files\Thesis\Cygnal_debug_log\FreeRTOSv202011.00\FreeRTOS\Demo\Cygnal>
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Source/list.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
../Source/portable/MemMang/heap_1.c:91: warning 127: non-pointer type cast to generic pointer
from type 'unsigned-long-int register'
to type 'unsigned-char generic* fixed'
../Source/portable/MemMang/heap_1.c:91: warning 127: non-pointer type cast to generic pointer
from type 'unsigned-long-int fixed'
to type 'unsigned-char generic* fixed'
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Source/portable/SDCC/Cygnal/port.c
../Source/portable/SDCC/Cygnal/port.c:96: error 1: Syntax error, declaration ignored at 'data'
../Source/portable/SDCC/Cygnal/port.c:60: syntax error: token -> 'xdata'; column 5
../Source/portable/SDCC/Cygnal/port.c:60: error 7: Old style C declaration. IGNORED 'ucStackBytes'
sdcc.exe: fatal error: when writing output to : Broken pipe
make: *** [../Source/portable/SDCC/Cygnal/port.rel] Error 1

```

Figure 3.32: error 1: Syntax error, declaration ignored at 'data'

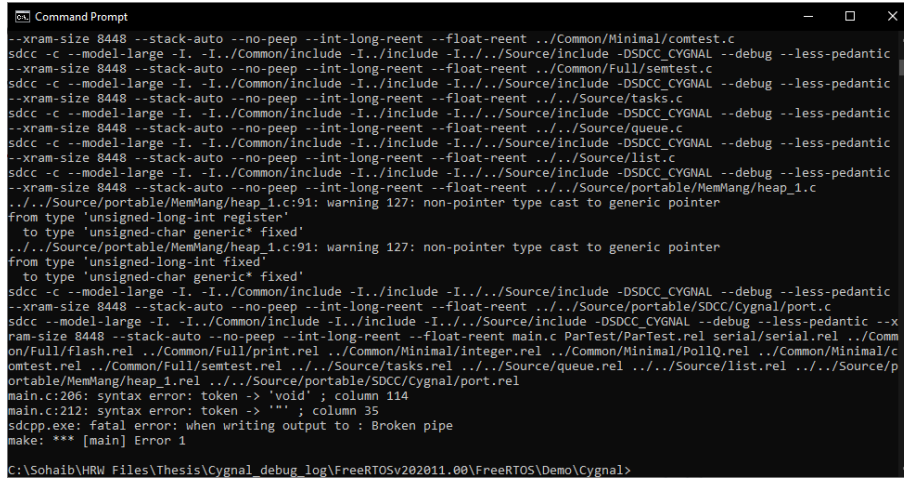
The keywords *"data"*, *"xdata"*, *"asm"*, *"endasm"*, *"naked"* and *"interrupt"* are no longer supported by any version of SDCC 2.4.0 and higher.

## Fix

All instances of the keywords in the file *"../FreeRTOS/Source/portable/SDCC/Cygnal/port.c"* must be changed to their updated format, *"\_\_data"*, *"\_\_xdata"*, *"\_\_asm"*, *"\_\_endasm"*, *"\_\_naked"* and *"\_\_interrupt"* respectively. This can be done using the Find and Replace functionality of most commonly available text editors.

### 3.2.12 Changelog #12 : main.c

#### Issue



```
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Minimal/comtest.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Full/semtest.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Source/tasks.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Source/queue.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Source/list.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Source/portable/MemMang/heap_1.c
../Source/portable/MemMang/heap_1.c:91: warning 127: non-pointer type cast to generic pointer
from type 'unsigned-long-int register'
to type 'unsigned-char generic* fixed'
../Source/portable/MemMang/heap_1.c:91: warning 127: non-pointer type cast to generic pointer
from type 'unsigned-long-int fixed'
to type 'unsigned-char generic* fixed'
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Source/portable/SDCC/Cygnal/port.c
sdcc --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic --x
ram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent main.c ParTest/ParTest.rel serial/serial.rel ../Comm
on/Full/flash.rel ../Common/Full/print.rel ../Common/Minimal/integer.rel ../Common/Minimal/PollQ.rel ../Common/Minimal/c
omtest.rel ../Common/Full/semtest.rel ../Source/tasks.rel ../Source/queue.rel ../Source/list.rel ../Source/p
ortable/MemMang/heap_1.rel ../Source/portable/SDCC/Cygnal/port.c
main.c:206: syntax error: token -> 'void' ; column 114
main.c:212: syntax error: token -> '''' ; column 35
sdcc.exe: fatal error: when writing output to : Broken pipe
make: *** [main] Error 1
C:\Sohaib\HRW Files\Thesis\Cygnal_debug_log\FreeRTOSv202011.00\FreeRTOS\Demo\Cygnal>
```

Figure 3.33: syntax error: token -> 'void'

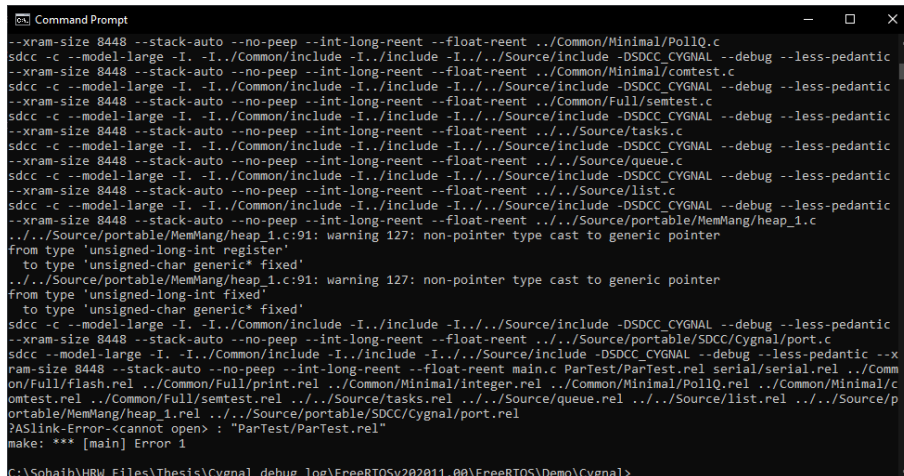
The keywords *"xdata"*, *"asm"* and *"endasm"* are no longer supported by any version of SDCC 2.4.0 and higher.

#### Fix

All instances of the keywords in the file *"../FreeRTOS/Demo/Cygnal/main.c"* must be changed to their updated format, *"\_\_xdata"*, *"\_\_asm"* and *"\_\_endasm"* respectively. This can be done using the Find and Replace functionality of most commonly available text editors.

### 3.2.13 Changelog #13 : Moving .rel and other files

#### Issue



```
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Minimal/PollQ.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Minimal/comtest.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Common/Full/semtest.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Source/tasks.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Source/queue.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Source/list.c
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
--xram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent ../Source/portable/MemMang/heap_1.c
../Source/portable/MemMang/heap_1.c:91: warning 127: non-pointer type cast to generic pointer
from type 'unsigned-long-int register'
to type 'unsigned-char generic* fixed'
../Source/portable/MemMang/heap_1.c:91: warning 127: non-pointer type cast to generic pointer
from type 'unsigned-long-int fixed'
to type 'unsigned-char generic* fixed'
sdcc -c --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic
sdcc --model-large -I. -I../Common/include -I../include -I../Source/include -DSDCC_CYGNAL --debug --less-pedantic --x
ram-size 8448 --stack-auto --no-peep --int-long-reent --float-reent main.c ParTest/ParTest.rel serial/serial.rel ../Comm
on/Full/flash.rel ../Common/Full/print.rel ../Common/Minimal/integer.rel ../Common/Minimal/PollQ.rel ../Common/Minimal/c
omtest.rel ../Common/Full/semtest.rel ../Source/tasks.rel ../Source/queue.rel ../Source/list.rel ../Source/p
ortable/MemMang/heap_1.rel ../Source/portable/SDCC/Cygnal/port.c
?ASLink-Error-<cannot open> : "ParTest/ParTest.rel"
make: *** [main] Error 1
C:\Sohaib\HRW Files\Thesis\Cygnal_debug_log\FreeRTOSv202011.00\FreeRTOS\Demo\Cygnal>
```

Figure 3.34: ?ASLink-Error-<cannot open> : "ParTest/ParTest.rel"

The error here is caused by an incorrect filepath provided by the default makefile. Once the SDCC compiles a source file, it generates a list of files, namely: the .adb, .asm, .lst, .rel and .sym files with the same filenames as the source file, however it does so in the default directory where the *"main.c"* file is located. According to the makefile, these files must be located in the same directory as the .c source file they were originally compiled from.

## Fix

**NOTE:** At this point, the code MUST be compiled atleast once to generate all relevant .rel and .asm files for each individual module.

Once this has been done, all 5 files (.adb, .asm, .lst, .rel and .sym) must be relocated to the appropriate directory per module (per source file). A table containing all the relevant directory paths is shown below:

Source File	Source Directory
flash.c	../FreeRTOSv202011/FreeRTOS/Demo/Common/Full/
print.c	../FreeRTOSv202011/FreeRTOS/Demo/Common/Full/
semtest.c	../FreeRTOSv202011/FreeRTOS/Demo/Common/Full/
integer.c	../FreeRTOSv202011/FreeRTOS/Demo/Common/Minimal/
PollQ.c	../FreeRTOSv202011/FreeRTOS/Demo/Common/Minimal/
comtest.c	../FreeRTOSv202011/FreeRTOS/Demo/Common/Minimal/
tasks.c	../FreeRTOSv202011/FreeRTOS/Source/
queue.c	../FreeRTOSv202011/FreeRTOS/Source/
list.c	../FreeRTOSv202011/FreeRTOS/Source/
ParTest.c	../FreeRTOSv202011/FreeRTOS/Demo/ParTest/
serial.c	../FreeRTOSv202011/FreeRTOS/Demo/serial/
heap_1.c	../FreeRTOSv202011/FreeRTOS/Source/portable/MemMang/
port.c	../FreeRTOSv202011/FreeRTOS/Source/portable/SDCC/Cygnal/

Table 3.1: Source files and their respective directories

## 3.3 packihx

After the generated files are relocated in the same directory as the source files, typing *"make"* one last time in the console should then generate an updated *"main.ihx"* file within the directory *"../FreeRTOS/Demo/Cygnal/"*. This file is then the hex code generated to be flashed onto an 8051f120 board. While most flash tools should be able to flash the ".ihx" file, it may be necessary to convert it to a ".hex" file format before flashing. This can be done on Windows using a program called *"packihx"*.

```
D:\SDCC-8051-tutorial\LedBlink>packihx LedBlink.ihx > LedBlink.hex
packihx: read 13 lines, wrote 17: OK.

D:\SDCC-8051-tutorial\LedBlink>ls
LedBlink.asm  LedBlink.lk  LedBlink.mem  LedBlink.sym
LedBlink.hex  LedBlink.lst  LedBlink.rel  compile.bat
LedBlink.ihx  LedBlink.map  LedBlink.rst  ledblink.c

D:\SDCC-8051-tutorial\LedBlink>
```

Figure 3.35: Using packihx to convert the .ihx file to a .hex file [13]

Simply open a command window and type *"packihx main.ihx > main.hex"* in the directory where the *"main.ihx"* file is located.

We end up with a ".hex" file ready to flash using any generic flash programmer available. However, before this, we must adapt the code to our particular hardware.

## Chapter 4

# Adapting the Code

### 4.1 Adapting the debugged Cygnal code for our Hardware

The next main issue has mainly to do with the fact that the original Cygnal Port was written for hardware (namely the 8051f120 microcontroller) that was different from the one I had for testing (The EFM8BB1 Starter Kit). The source Cygnal code was using Port 3 (P3) to access an array of 8 LEDs and Port 1 (P1) to access one onboard LED, and my hardware did not possess the matching ports. Not only this, but according to the author of the Cygnal port [14], some features in the 8051f120 were taken directly from an 8052 standard architecture and were not available in the default 8051, as seen in Figure 4.1.

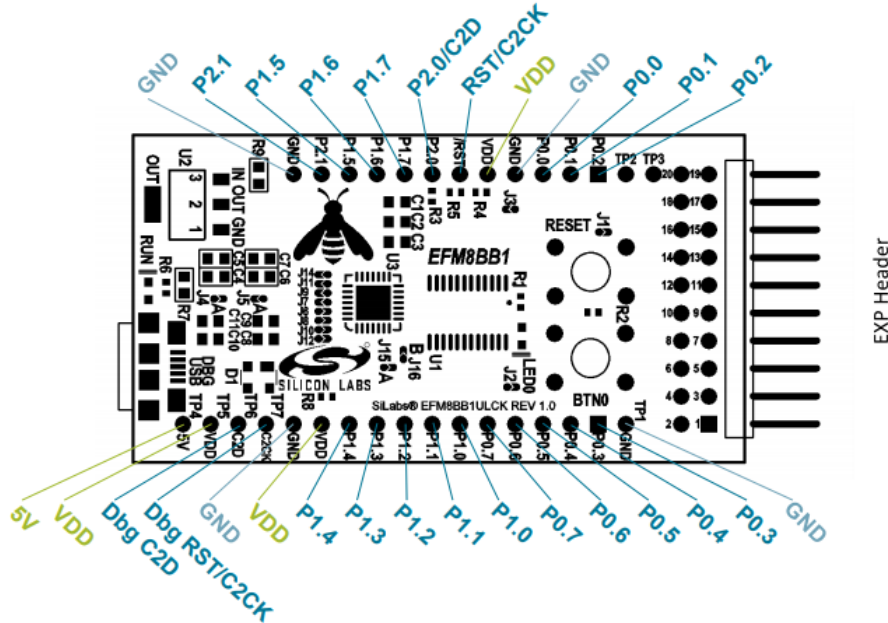


Figure 4.1: Pinout for the EFM8BB1 [15] (standard 8051 architecture)

To get around this problem, I first had to ensure that both hardware were using the same timer for generating the FreeRTOS scheduler tick interrupt. This was indeed the case as both hardware had access to Timer2 which is by default a 16Bit timer. Next I had to ensure that various important IO control registers were addressed by the same memory location in both the 8051f120 and a standard 8051 architecture. This can be done by checking the file `../FreeRTOS/Demo/Cygnal/c8051f120.h` and comparing the addresses of the `XBR2`, `P1MDOUT`, and `P1MDIN` registers to the Reference Manual of the EFM8BB1 [16]. Once I had confirmed that all registers were addressed correctly, I then needed to modify the code one more time to now use P1 for an array of LEDs (instead of P3) and to use a single bit of P2 for the additional on-board LED (instead of P1).

The following changes needed to be made in the `"../FreeRTOS/Demo/Cygnal/ParTest/ParTest.c"` and the `"../FreeRTOS/Demo/Cygnal/main.c"` files:

Initially, within the file `"../FreeRTOS/Demo/Cygnal/ParTest/ParTest.c"`, the P3 outputs must be disabled and instead the outputs for P1 should be enabled for the LED array. Simply comment out the required lines within the `"vParTestInitialise"` function to use the P1 port for accessing the LED array.

NOTE: The sourcefile contains a comment here which implies that activating P3 as push-pull is done for the onboard LED. This is incorrect and is actually done to use the LED array.

```

50 void vParTestInitialise( void )
51 {
52     unsigned char ucOriginalSFRPage;
53
54     /* Remember the SFR page before it is changed so it can get set back
55     before the function exits. */
56     ucOriginalSFRPage = SFRPAGE;
57
58     /* Setup the SFR page to access the config SFR's. */
59     SFRPAGE = CONFIG_PAGE;
60
61     /* Set the on board LED to push pull. */
62     //P3MDOUT |= partstPUSH_PULL;
63     P1MDOUT |= partstPUSH_PULL;
64
65     /* Return the SFR page. */
66     SFRPAGE = ucOriginalSFRPage;
67
68     //P3 = partstALL_OUTPUTS_OFF;
69     P1 = partstALL_OUTPUTS_OFF;
70 }
71 /*-----*/

```

Figure 4.2: P3 outputs disabled and P1 outputs enabled (ParTest.c)

Next, within the functions `"vParTestSetLED"` and `"vParTestToggleLED"` all instances of "P3" must be replaced with "P1" as we are no longer have and LED array on P3 and instead need to use P1.

```

73 void vParTestSetLED( unsigned portBASE_TYPE uxLED, signed portBASE_TYPE xValue )
74 {
75     portBASE_TYPE xError = pdFALSE;
76
77     vTaskSuspendAll();
78     {
79         if( xValue == pdFALSE )
80         {
81             switch( uxLED )
82             {
83                 case 0 : P1 |= partstOUTPUT_0;
84                         break;
85                 case 1 : P1 |= partstOUTPUT_1;
86                         break;
87                 case 2 : P1 |= partstOUTPUT_2;
88                         break;
89                 case 3 : P1 |= partstOUTPUT_3;
90                         break;
91                 case 4 : P1 |= partstOUTPUT_4;
92                         break;
93                 case 5 : P1 |= partstOUTPUT_5;
94                         break;
95                 case 6 : P1 |= partstOUTPUT_6;
96                         break;
97                 case 7 : P1 |= partstOUTPUT_7;
98                         break;
99                 default : /* There are no other LED's wired in. */
100                          xError = pdTRUE;
101                          break;

```

Figure 4.3: P3 outputs replaced by P1 outputs (1) (ParTest.c)

```

164
165     if( xError != pdTRUE )
166     {
167         if( P1 & ucBit )
168         {
169             P1 &= ~ucBit;
170         }
171         else
172         {
173             P1 |= ucBit;
174         }
175     }
176 }
177 xTaskResumeAll();
178 }
179

```

Figure 4.4: P3 outputs replaced by P1 outputs (2) (ParTest.c)

Now within the file `"../FreeRTOS/Demo/Cygnal/main.c"`, the output to Port 1 must be changed to Port 2 as we no longer have enough IO pins on Port 1 after switching the LED Array outputs here. To do this, firstly, some variables need to be defined for the particular Pin we wish to use on Port 2.

```

89  /* Constants to setup and use the on board LED. */
90  #define ucLED_BIT          ( ( unsigned char ) 0x40 )
91  #define my_ucLED_BIT       ( ( unsigned char ) 0x01 )
92  #define mainPORT_1_BIT_6   ( ( unsigned char ) 0x40 )
93  #define my_mainPORT_2_BIT_0 ( ( unsigned char ) 0x01 )
94  #define mainENABLE_CROSS_BAR ( ( unsigned char ) 0x40 )
95

```

Figure 4.5: Define the bit position relative to Port 2 as `"my_ ..."` variables (main.c)

Next, we need to enable Port 2 as an output push-pull using the Port bit definition declared earlier (`"my_mainPORT_2_BIT_0"`) and comment out the previous code for enabling Port 1 in `"main.c"` as this has already been done in `"ParTest.c"`.

```

244  /* Disable the watchdog. */
245  WDTCN = mainDISABLE_BYTE_1;
246  WDTCN = mainDISABLE_BYTE_2;
247
248  /* Set the on board LED to push pull. */
249  //P1MDOUT |= mainPORT_1_BIT_6;
250  P2MDOUT |= my_mainPORT_2_BIT_0;
251
252  /* Setup the cross bar to enable serial comms here as it is not part of the
253  standard 8051 setup and therefore is not in the driver code. */
254  XBR0 |= mainENABLE_COMS;
255  P0MDOUT |= mainCOMS_LINES_TO_PUSH_PULL;

```

Figure 4.6: Enable Port 2 outputs as Push-Pull (main.c)



Finally we need to change the function *prvToggleOnBoardLED* to now output to Port 2 when toggling the LED. Simply copy and comment out the previous lines of code, and change all instances of P1 to P2, while also replacing the variable *ucLED\_Bit* with *my\_ucLED\_Bit*.

```
323 static void prvToggleOnBoardLED( void )
324 {
325     /* If the on board LED is on, turn it off and vice versa. */
326     // if( P1 & ucLED_BIT )
327     // {
328     //     P1 &= ~ucLED_BIT;
329     // }
330     // else
331     // {
332     //     P1 |= ucLED_BIT;
333     // }
334     // }
335
336     if( P2 & my_ucLED_BIT )
337     {
338         P2 &= ~my_ucLED_BIT;
339     }
340     else
341     {
342         P2 |= my_ucLED_BIT;
343     }
344 }
345 /*-----*/
```

Figure 4.7: Setting outputs of *prvToggleOnBoardLED* function to P2 (main.c)

Once all the changes have been made, opening a console window and typing *make* in the same directory as the Makefile, will once again generate files relating to *ParTest.c* as mentioned in Changelog # 13 (3.2.13) and as such, these files will need to be relocated and must replace the files where the original *ParTest.c* sourcefile is located. Similarly, the generated .ihx file must once again be converted to a .hex file using packihx so that we may flash it.

## Chapter 5

# Limitations

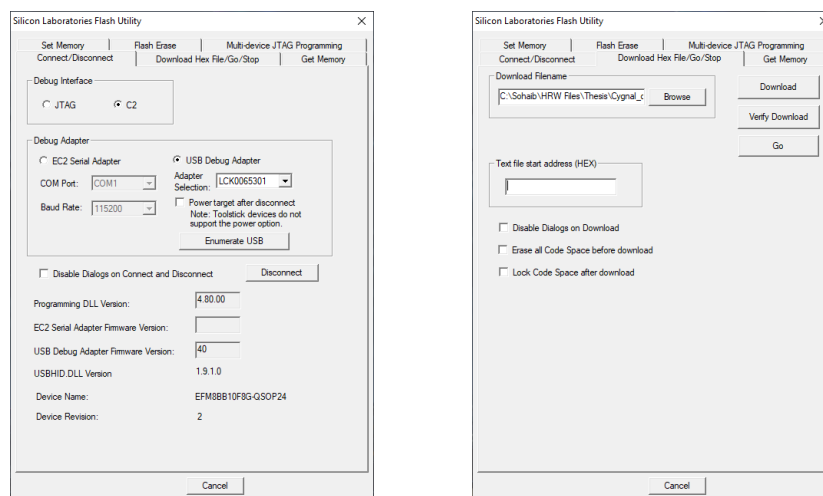
### 5.1 Silicon Labs IDE

The Cygnal Port guide on the FreeRTOS page [14] mentioned the use of the Silicon Labs IDE [17] when going further with trying to get a running example of the Cygnal Port. This was attempted, however it led to some unexpected errors that were not present so far during code compilation. Initially, the *"sdcc.wsp"* project file for this port needed to be edited in a text editor to correctly include the source file directories, but failed when attempting to *"#include"* files within the code. To fix this issue, the full file paths needed to be provided when using the *#include* keyword in the code and despite these changes, when using the SDCC standard *"2.x.x"* to compile the code there were still errors regarding RAM size.

To avoid these unnecessary errors, I switched to simply uploading the generated *.hex* file that I got from the Makefile.

### 5.2 Testing the Debugged Cygnal Port

According to the Cygnal Port's guide on the FreeRTOS page [14], it was recommended to use the Silicon Labs Flash Programmer which was installed with the Silicon Labs IDE. As seen in Figure 5.1, using the Silicon Labs Flash Programming Utility, we can connect to our EFM8BB1 board and attempt to flash the *.hex* file located in the same directory as the *main.c* file.



(a) Connecting to the Board

(b) Downloading the updated *.hex* file

Figure 5.1: Using the Silicon Labs Flash Programming Utility

Unfortunately, flashing the code on the real 8051 hardware returned an error I did not have the time or knowledge to fix within the timeframe of the Bachelor thesis. An educated guess for the error is that there is either not enough external RAM within our hardware setup as compared to the 8051f120 board or that the external RAM is addressed differently in the Cygnal Port source code and within my hardware setup. The Silicon Labs Flash Utility does not return a very helpful error message to help debug this issue, as seen in Figure 5.2

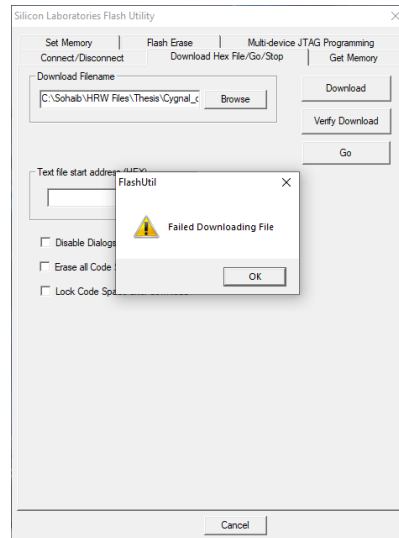


Figure 5.2: Silicon Labs Flash Programming Utility Error message

So naturally, my next best attempt to test the debugged code was to use a Free online 8051 simulator known as "EDSIM51" [18]. This simulator aims to accurately replicate the entire architecture of the original Intel 8051 Microcontroller and as such was an excellent alternative to use for my testing needs. However, this too gave rise to new issues when trying to upload my .hex file to the software. Initially, it simply could not run the Assembler file generated from the sourcecode due to an unknown error. After that I tried dumping the .hex file directly to the software and while it did appear to run, EDSIM threw a warning for the MOVX command used within the assembler code as seen in Figure 5.3

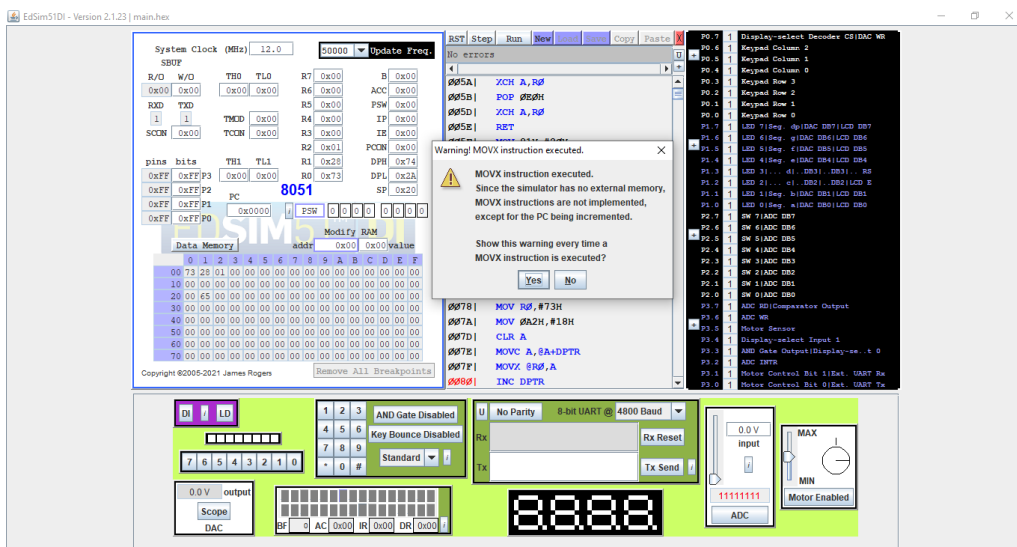


Figure 5.3: EDSIM51 MOVX warning message

This warning is caused by the use of the command "MOVX" within the source code for the FreeRTOS Cygnal Port. Since the simulator cannot have physical external memory or RAM, it merely interprets the MOVX command as an incrementor to the current Program Counter (PC)

value. As such, any context switching action might be misinterpreted by the simulator and I do not know any alternatives to fix this issue. Additionally, if this warning is ignored, the code appears to run, but I could not confirm correct operation as the LED array provided within the EDSIM51 software did not show any changes as the code ran. I do not know if this issue is caused by the MOVX warning or if there is a logical error in the source code.

## Chapter 6

# Conclusion

Regrettably, I could not manage to achieve the initial goals of this Bachelor Thesis. However it is my understanding that by initiating some work on debugging the long overdue issues with the FreeRTOS Cygnal Port, it may be possible one day to successfully test on real hardware, and gain a working model which can allow us to get a closer detailed look at FreeRTOS.

It is my hope that one day FreeRTOS will become an incredibly easy piece of software to use and implement within all applications to allow for parallel, deterministic and debug-friendly code execution.

# Bibliography

- [1] **Github Repository.** [https://github.com/SohaibSaeed26/8051FPGA\\_CygnalCore](https://github.com/SohaibSaeed26/8051FPGA_CygnalCore). Accessed: 23.02.2021.
- [2] **FreeRTOS Source Code** download center. <https://www.freertos.org/a00104.html>. Accessed: 04.01.2021.
- [3] **FreeRTOS Quickstart Guide.** <https://www.freertos.org/FreeRTOS-quick-start-guide.html>. Accessed: 06.01.2021.
- [4] **Oregano Systems 8051 IP Core** product page. <https://www.oreganosystems.at/products/ip-cores/8051-ip-core>. Accessed: 07.01.2021.
- [5] **Quartus Prime Lite IDE** download page. <https://fpgasoftware.intel.com/20.1/?edition=lite&platform=windows>. Accessed: 07.01.2021.
- [6] **Small Device C Compiler** home page. <http://sdcc.sourceforge.net/>. Accessed: 08.01.2021.
- [7] **GNU Make** download page. <https://www.gnu.org/software/make/>. Accessed: 08.01.2021.
- [8] **Cyclone 10 CYC1000** product page. <https://shop.trenz-electronic.de/en/TEI0003-02-CYC1000-with-Cyclone-10-FPGA-8-MByte-SDRAM>. Accessed: 10.01.2021.
- [9] **Busy Bee 8051 Development Kit** prodcut page. <https://www.silabs.com/development-tools/mcu/8-bit/efm8bb1lck-starter-kit>. Accessed: 12.01.2021.
- [10] **FreeRTOS Defined** about freertos. <https://www.freertos.org/about-RTOS.html>. Accessed: 06.01.2021.
- [11] **FreeRTOS Implementation** about freertos. <https://www.freertos.org/implementation/a00004.html>. Accessed: 06.01.2021.
- [12] **SDCC include print.h bug.** <https://sourceforge.net/p/sdcc/bugs/2629/>. Accessed: 24.01.2021.
- [13] **8051 Software Development using SDCC.** <https://www.xanthium.in/programming-8051-8052-using-opensource-small-device-c-compiler-tutorial>. Accessed: 26.01.2021.
- [14] **FreeRTOS Cygnal Port Homepage.** <https://www.freertos.org/portcygn.html#ConfigAndUsage>. Accessed: 20.01.2021.
- [15] **Busy Bee 8051 Development Kit** user guide. <https://www.silabs.com/documents/public/user-guides/ug377-efm8bb1lck.pdf>. Accessed: 12.01.2021.
- [16] **Busy Bee 8051 Development Kit** datasheet. <https://www.silabs.com/documents/public/reference-manuals/efm8bb1-rm.pdf>. Accessed: 12.01.2021.
- [17] **Silicon Labs IDE** download page. <https://www.silabs.com/developers/8-bit-8051-microcontroller-software-studio>. Accessed: 28.01.2021.
- [18] **The 8051 Simulator for Teachers and Students.** <https://www.edsim51.com/index.html>. Accessed: 01.02.2021.