Astro CSoC 8051 code and xdata memory banking

Content

Introduction

This document describes the use of 8051 code and xdata banking based on AagteLogic Astro Configurable System-on-Chip(CSoC). It explains the settings needed to generate the proper code (using the Keil C Compiler) for code and xdata banking and the steps to download the code to Astro CSoC using Agdi (Astro CSoC 8051 debugger).

Code and xdata memory banking

Astro CSoC 8051 supports the code/xdata memory banking in the memory configuration. Astro CSoC 8051 incorporates the Harvard architecture, with separate code and data spaces. Memory organization is similar to that of the industry standard 8051. There are three memory areas: Code Memory (Internal ROM or External ROM), External Data Memory(xdata, External RAM) and Internal Data Memory(Internal RAM).

The Astro CSoC 8051 can be optionally configured to extend both code memory and external data memory(independently) up to 8MB by means of dedicated 8-bit page address register.

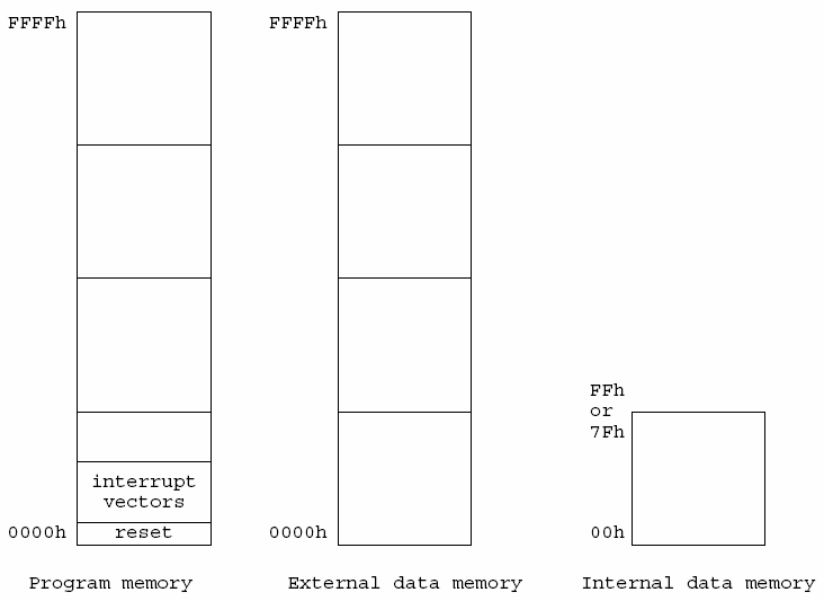

Figure 1 Astro CSoC 8051 memory map

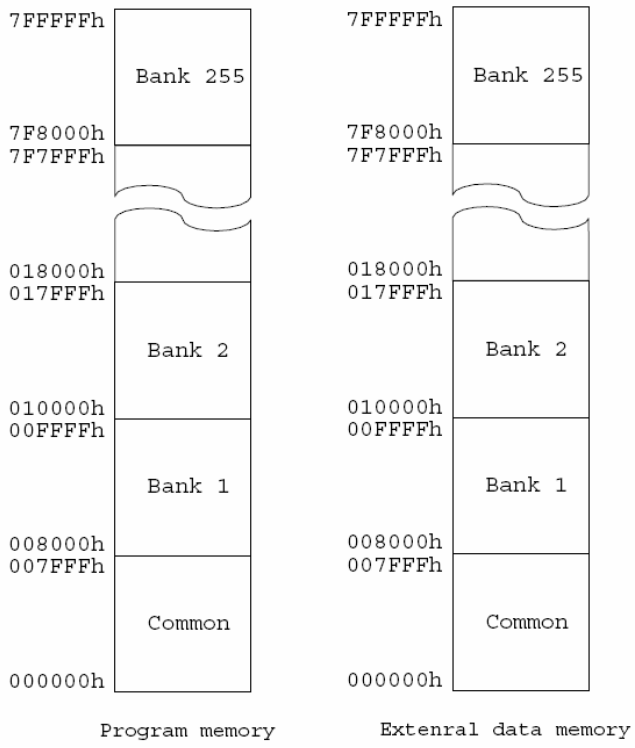

Figure 2 Astro CSoC 8051 memory map

As is shown in figure 2, the common area starts at 0x0000H address and ends at 0x7FFFH. Bank 0 is not available, it is physically the same memory spaces at the common area.

While code/xdata memory banking feature is enabled, the code/xdata memory address is composed of two parts:

- 15 bit address
- 8bits from a bank switching register SW_REGISTER:

- PAGESEL for code memory
- D_PAGESEL for xdata memory

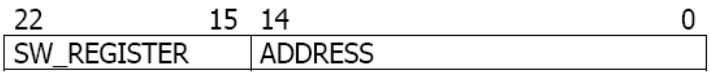

Figure 3 code/xdata memory address with banking

For more details, please refer to *Astro Device Data Sheet*

Code memory banking – Keil uVision compilation options

Choose the **Project->Options** from the Keil uVision main menu. On the section **Target** please check the **Code Banking** option and set the number of code banks, four for example. Next, set the common and bank memory space. On the figure below, the common space is set at a range from 0x0000 to 0x7FFF, the CODE banks area is set from 0x8000 to 0xFFFF. The Keil uVision2/3 supports up to **64** banks. Allowed values are: **2**, **4**, **8**, **16**, **32** or **64**.

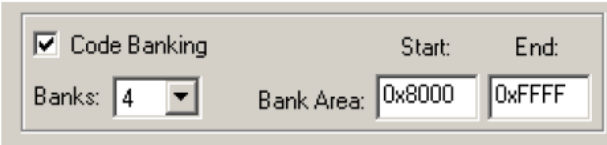

Figure 4 code banking setup

Furthermore, the options for target should be set as follows:

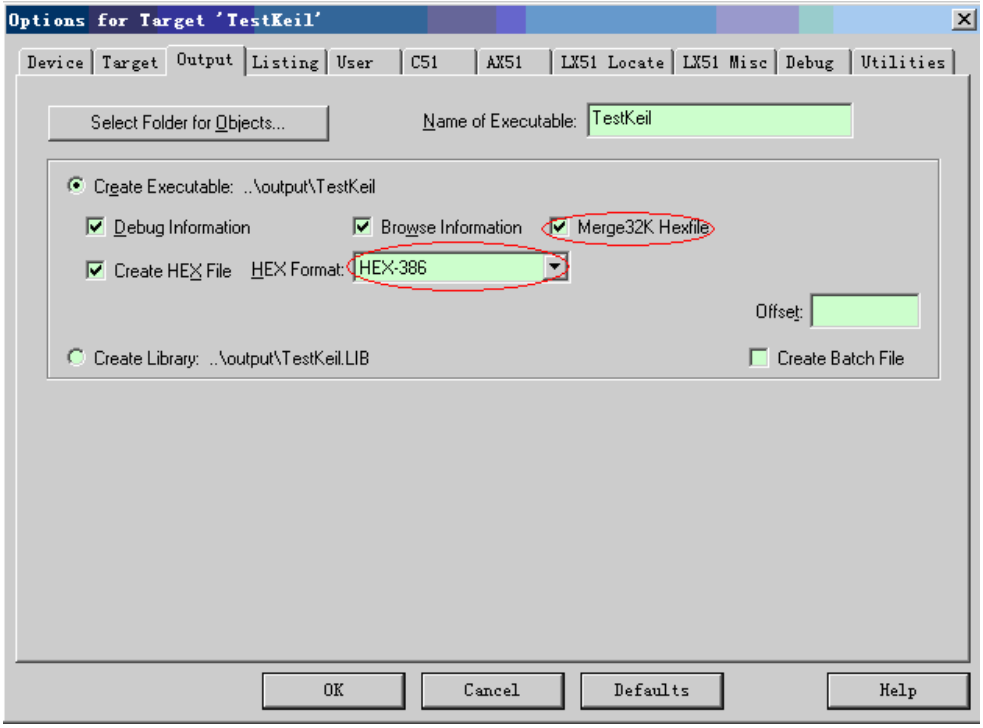

Figure 5 code banking options for target setting

For Astro CSoC 8051 only supports code banking with 32KB, so here suggest you select **Merge 32K Hexfile** option and the HEX formant should be set as **HEX-386**

The Astro CSoC 8051 banking mechanism is similar to the example hardware documented at http://www.keil.com/support/man/docs/lx51/lx51_bk_example4.htm. For the Astro CSoC 8051, user should never assign the Bank 0 for any of the project file! Bank 0 is the common area. The <default> setting is not allowed too. The main difference between the documented example and the Astro 8051 banking mechanism is that the PAGESEL register is used instead of port P1.

All you need to do are some simple modifications to **L51_BANK.A51** and **STARTUP.A51** files. You can find these files in the Keil uVision directory (i.e.: Keil\C51\lib). Copy these files and add to your project workspace. Next modify as shown below.

For the Astro CSoC 8051 the configuration file (L51_BANK.A51) should be modified as follows: B_NBANKS is the number of banks, can be modified according to your requirements. B_MODE should be set 4, using user-provided bank switch code. B_VAR_BANKING should be set 0, variable banking dose not use L51_BANK.51.

```
?B_NBANKS       EQU  4     ; Define maximum Number of Banks
;                          ; following values are allowed: 2, 4, 8, 16, 32, 64
;                          ; for BL51 the maximum value for ?B_NBANKS is 32
;                          ; for LX51 the maximum value for ?B_NBANKS is 64
;
; <o> ?B_MODE: Bank Switching via
;                     <0=> 8051 Port
;                     <1=> XDATA Port
;                     <4=> User-provided bank switch code
?B_MODE         EQU  4     ; 0 for Bank-Switching via 8051 Port
;                          ; 1 for Bank-Switching via XDATA Port
;                          ; 4 for user-provided bank switch code
;
; <q> ?B_RTX: The application uses RTX-51 Real-time OS
?B_RTX          EQU  0     ; 0 for applications without real-time OS
;                          ; 1 for applications using the RTX-51 real-time OS
;
; <q> ?B_VAR_BANKING: Variable banking uses this L51_BANK module
;     <i> Notes: ?B_VAR_BANKING uses the 'far' and 'far const' C51 memory types to
;     <i>         extent the space for variables in RAM and/or ROM of classic 8051
;     <i>         device.  The same hardware as for code banking is used.  Program
;     <i>         code banking and variable banking share the same hardware I/O pins.
;     <i>         The C51 Compiler must be used with the VARBANKING directive.
;     <i>         Variable Banking is only supported with the LX51 linker/locater.
?B_VAR_BANKING  EQU  0     ; Variable Banking via L51_BANK (far memory support)
;                          ; 0 Variable Banking does not use L51_BANK.A51
;                          ; 1 Variable Banking uses this L51_BANK.A51 module
```

Figure 6 code banking setting in L51_BANK.A51

Additionally (for the same file L51_BANK.A51), modify the bank switch code procedure:

```
IF  ?B_MODE = 4;
.........
PAGESEL         DATA 94H ; PAGESEL register
SWITCH0         MACRO
                MOV PAGESEL, #1
                ENDM
SWITCH1         MACRO
                MOV PAGESEL, #1
                ENDM
SWITCH2         MACRO
                MOV PAGESEL, #2
                ENDM
SWITCH3         MACRO
                MOV PAGESEL, #3
                ENDM

ENDIF;
```

Figure 7 code banking switch code setting in L51_BANK.A51

In the STARTUP.A51 file, delete #if and #endif directives at the end of file.

```
; This code is required if you use L51_BANK.A51 with Banking Mode 4
;<h> Code Banking
; <q> Select Bank 0 for L51_BANK.A51 Mode 4
;#if 0
;     <i> Initialize bank mechanism to code bank 0 when using L51_BANK.A51 with Banking Mode 4.
EXTRN CODE (?B_SWITCH0)
                CALL    ?B_SWITCH0      ; init bank mechanism to code bank 0
;#endif
;</h>
                LJMP    ?C_START

                END
```

Figure 8 code banking setting in STARTUP.A51

Now, you can set a bank number for the chosen file. To do it, select the chosen file in the **Project Workspace** and open its options window:

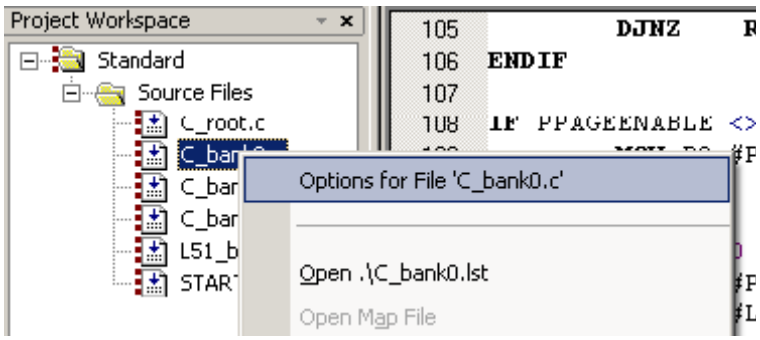

Figure 9 option window for project file

Next, set the bank number for chosen file.

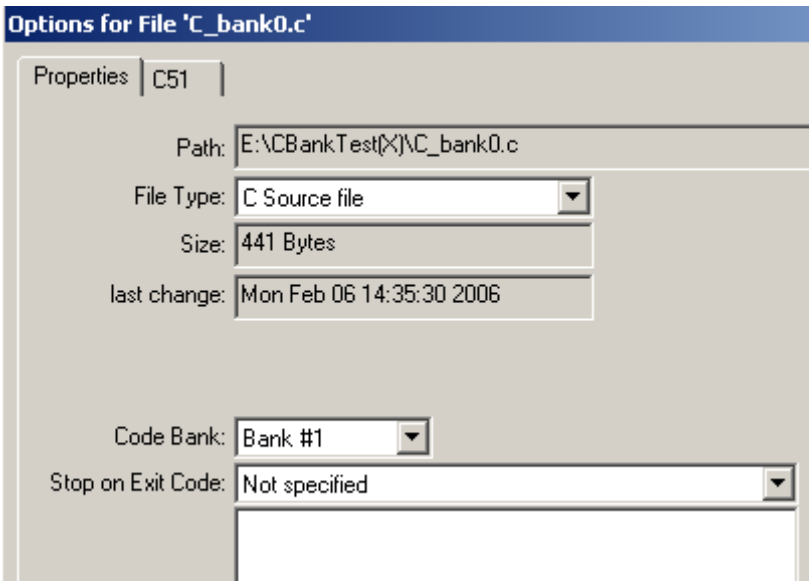

Figure 10 bank number field

NOTE: for the Astro CSoC 8051, user should never assign the Bank #0 for any of the project file! The <default> setting is not allowed too.

Furthermore, Astro CSoC 8051 debugger also Agdi support code banking. You can download code with banking through Agdi and check whether download code is correct through memory window, as is shown in figure 11.

For the bank size of Astor CSoC 8051 is 32K, Keil treat all bank size as 64K, so if you want to check bank1's code, you should set address as B1:0x8000, if set address as B1:0x0000, the code shown in memory window is the code that in common area. So if you want to check bank code, please set address as Bx:0x8000 (x is bank number).

Step and breakpoint is also supported by Agdi when code banking is used.

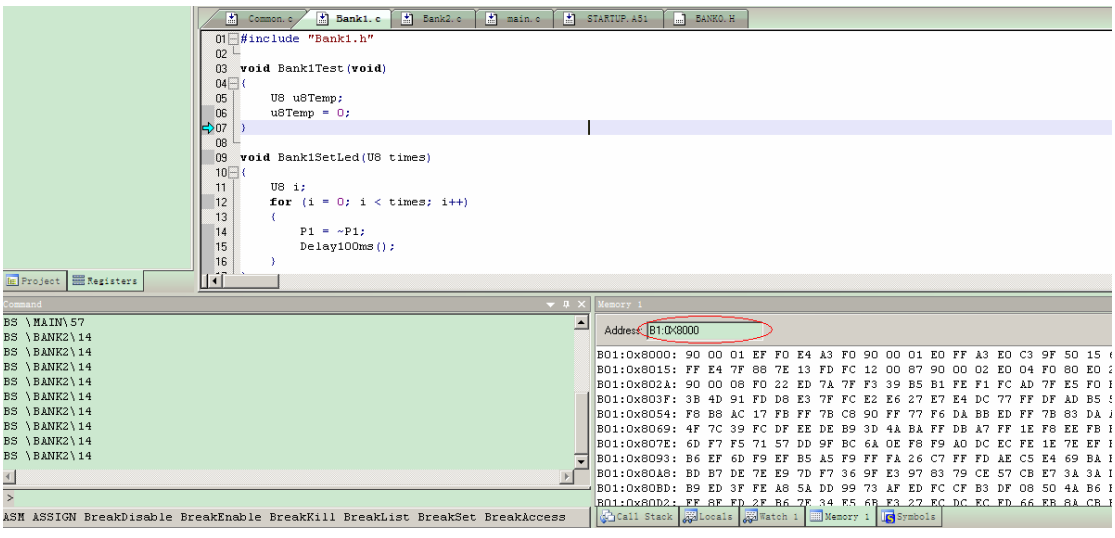

Figure 11 check bank code with memory window

Xdata memory banking – Keil uVision compilation options

The Agdi also supports the XDATA memory banking feature, this way user can read or write more than 64kB of XDATA memory. The Astro CSoC 8051 supports XDATA banking with 32KB pages. The SFR named D_PAGESEL is used to select the XDATA page. Since the Astro CSoC memory is fragmented into 32KB blocks, it is not possible to create arrays larger than 32KB.

Exact configuration of the XDATA memory banking is very well documented at Keil uVision Technical Support web
pages:

- http://www.keil.com/support/docs/2103.htm
- http://www.keil.com/support/man/docs/c51/c51_ap_xbanking.htm

You can find completely documented, ready-made example of the XDATA banking - in the Keil uVision installation directory: **Keil\C51\Examples\R8051XC\XBanking**.

For Astro CSoC 8051 xdata banking, you may need to note that: When the 8051 targets addresses between 0000h and 7FFFh, the additional bits of address bus are always 0. When the 8051 addresses higher than 7FFFh of the Program Memory, the 8-bit contents of the D_PAGESEL register is placed onto the “memaddr[22:15]” bits. The size of D_PAGESEL register can be chosen between 2 to 8 bits during core configuration (the size of “memaddr” bus is then 17 to 23 bits, respectively). The D_PAGESEL register of 1 bit only is not possible, since there would be no change in “memaddr” length from the standard option (16 bits, no banking). The maximum number of pages is 256 (the common area at 0-32kB, and 255 pages (banks) logically visible at addresses between 32kB-64kB).

Note that the 0 value of D_PAGESEL register should not be used since it leads to accessing the same physical area at logical address space 8000h-FFFFh as 0000h-7FFFh. In such case, the banked area overlaps with the common bank.

As is shown in figure 12 is a code example for Astor CSoC 8051 xdata banking, you can take it as a reference.

```
U8 xdata *pu8XdataStart = 0;
for (i = START_ADDR; i < EDN_ADDR; i++)
{
    page_index = i >> 15;
    D_PAGESEL = page_index; //set D_PAGESEL for xdata banking

    page_addr = i & 0x7FFF;
    if (page_index > 0)
    {
       page_addr = (i & 0x7FFF) | 0x8000; //set specific address in every bank
    }

    pu8XdataStart[page_addr] = user_data;
}
```

Figure 12 code example for xdata banking

Summary

Code banking is a very useful tool for loading large amounts of code into the Astro CSoC. It is necessary if the code is greater than 64KB. The Astor CSoC Agdi can help you to make this process simple and nearly glue-less.

Reference Document

Astro CSoC 8051 debugger Agdi user guide.pdf
Primace User Guide.pdf
Astro Device Datasheet.pdf
Astro 8051 Load Code from spi flash and Run in SRAM in AS Mode User Guide.pdf

About Capital Microelectronics

Capital Microelectronics is the global pioneer and leader of the innovative Adaptable Programmable Gate Array (APGA) technologies. The company offers a full spectrum of programmable logic devices, software design tools, intellectual property (IP) and design services. Focusing on multiple applications such as telecommunication equipments, industrial control systems and consumer products, we use the Chinese leading foundry partner, SMIC, to manufacture our chips to offer solutions tailored for the market in China.

Technical Support Assistance

Tel: +86 10 82150100

E-mail: support@capital-micro.com

Website: www.capital-micro.com

Inc. All trademarks are the property of their respective companies.