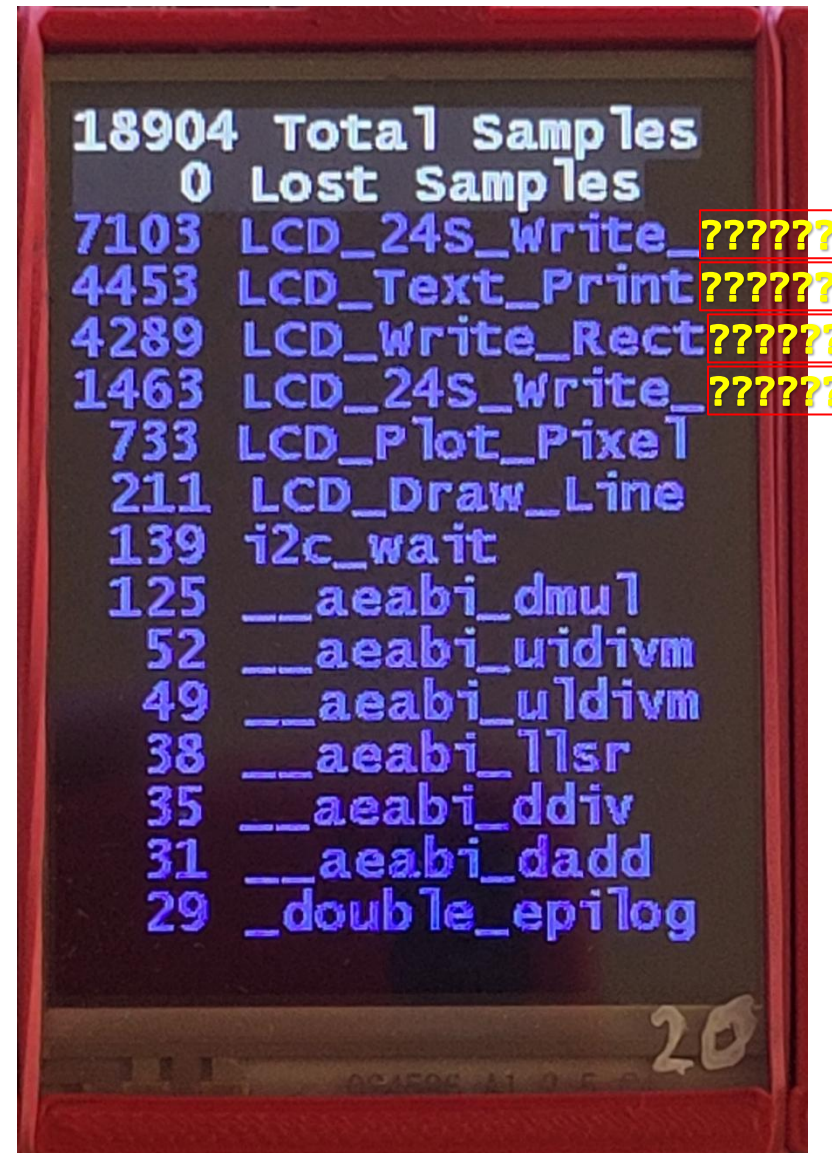


# Profiler – Resolving Function Name Ambiguity

2/14/2025

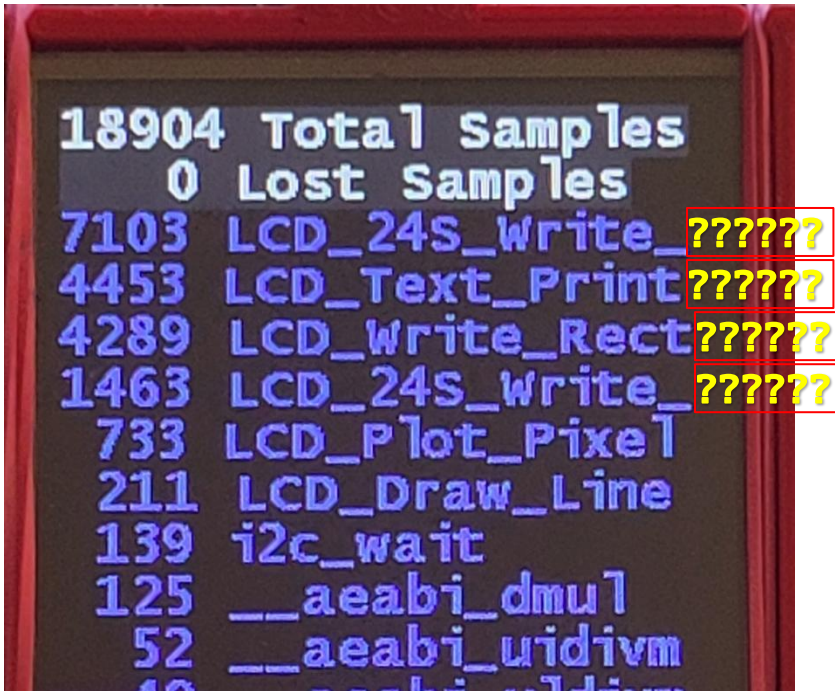
## Which Function Is It?



18904	Total Samples	
0	Lost Samples	
7103	LCD_24S_Write_	??????
4453	LCD_Text_Print	??????
4289	LCD_Write_Rect	??????
1463	LCD_24S_Write_	??????
733	LCD_Plot_Pixel	
211	LCD_Draw_Line	
139	i2c_wait	
125	__aeabi_dmul	
52	__aeabi_uidivm	
49	__aeabi_udivm	
38	__aeabi_llsr	
35	__aeabi_ddiv	
31	__aeabi_dadd	
29	_double_epilog	

20

# Two Reasons for Region Name Collisions from Truncation



- LCD size too small for font size
  - Long function names may go off-screen
  - LCD plotting functions don't print off-screen pixels

```
typedef struct {
    unsigned int Start;
    unsigned int End;
    char Name[24];
} REGION_T;

const REGION_T RegionTable[] = {
    {0x000000d5, 0x0000014c, "SystemInit"}, // 0
    {0x0000014d, 0x0000022c, "SystemCoreClockUpdate"}, // 1
    {0x00000261, 0x00000268, "Reset_Handler"}, // 2
    {0x00000269, 0x0000026a, "NMI_Handler"}, // 3
}
```

- RegionTable's Name field too short
  - REGION\_T uses 24 characters for Name to save space
  - Longer function names are truncated to fit
    - *Note to self: Discuss during Memory Size Optimization*
  - Further context
    - Each entry in profile table uses up this space, so space requirements grow very quickly. Usually >100 regions in program.
    - Can pass arguments to GetRegions.exe (edit script update\_regions\_MDK.cmd) to exclude specific functions from RegionTable (see documentation)

# Finding (“Disambiguating”) the Real Function Name

The LCD display shows the following data:

Count	Function Name
18904	Total Samples
0	Lost Samples
7103	LCD_24S_Write_
4453	LCD_Text_Print
4289	LCD_Write_Rect
1463	LCD_24S_Write_
733	LCD_Plot_Pixel
211	LCD_Draw_Line
139	i2c_wait
125	__aeabi_dmul
52	__aeabi_uidivm

The 'Watch 1' window shows the following data:

Name	Value
SortedRegions	0x1FFFF380
[0]	57
[1]	73
[2]	76
[3]	56
[4]	66
[5]	60
[6]	91
[7]	19
[8]	29
[9]	31

The 'RegionTable' window shows the following data:

Name	Value
RegionTable[SortedRegions[0]]	0x00004A64
Start	0x00000F85
End	0x00000FB8
Name	0x00004A6C "LCD_24S_Write_Data"
RegionTable[SortedRegions[1]]	0x00004C64
Start	0x00001DED
End	0x00001F4C
Name	0x00004C6C "LCD_Text_PrintChar"
RegionTable[SortedRegions[2]]	0x00004CC4
Start	0x00002035
End	0x00002086
Name	0x00004CCC "LCD_Write_Rectangle_Pixe"

- Load program into debugger
- Run program until sorted profile is printed on LCD
- Break program execution
- Open a Watch window
- Find element number **n** of truncated region name
  - Region names on LCD and SortedRegions array elements are in same order, starting with 0
  - This example: 0, 1, 2, 3
- Optional: add a watch for SortedRegions
- Add watch for element number **n**: RegionTable[SortedRegions[n]]
  - Expand to see fields, especially Name
  - If Name not truncated, then done.
  - What if Name is or might be truncated?  
LCD\_Write\_Rectangle\_Pixe



# Name Truncation in RegionTable? Option 1

The screenshot illustrates the process of finding source code for a specific address. The Watch window on the left shows a list of RegionTable entries. A 'Show Code at Address' dialog box is open, with the address 0x00002035 entered. A green dashed arrow points from this address to the Disassembly window, which shows the instruction 'STRH r5, [r6, #0x24]'. Another green dashed arrow points from the same address to the Source Code window, which shows the function 'void LCD\_Write\_Rectangle\_Pixel'.

Name	Value
RegionTable[SortedRegions[0]]	0x00004A64
Start	0x00000F85
End	0x00000FB8
Name	0x00004A6C "LCD_..."
RegionTable[SortedRegions[1]]	0x00004C64
Start	0x00001DED
End	0x00001F4C
Name	0x00004C6C "LCD_Text_PrintChar"
RegionTable[SortedRegions[2]]	0x00004CC4
Start	0x00002035
End	0x00002086
Name	0x00004CCC "LCD_Write_Rectangle_Pixe"

Disassembly:

```

0x00002035 84B5 STRH r5, [r6, #0x24]
0x00002037 03B0 LSLS r0, r6, #14
0x00002039 0290 LSLS r0, r2, #10
0x0000203B 0391 LSLS r1, r2, #14
0x0000203D 0899 LSRS r1, r3, #2
0x0000203F F8781022 DCD 0x1022F878
0x00002043 4940 LDR r1, [pc, #256] ; @0x00002144
0x00002045 4978 LDR r1, [pc, #480] ; @0x00002228
0x00002047 4009 ANDS r1, r1, r1
0x00002049 0118 TSTS r0, r3, #4
  
```

Source Code:

```

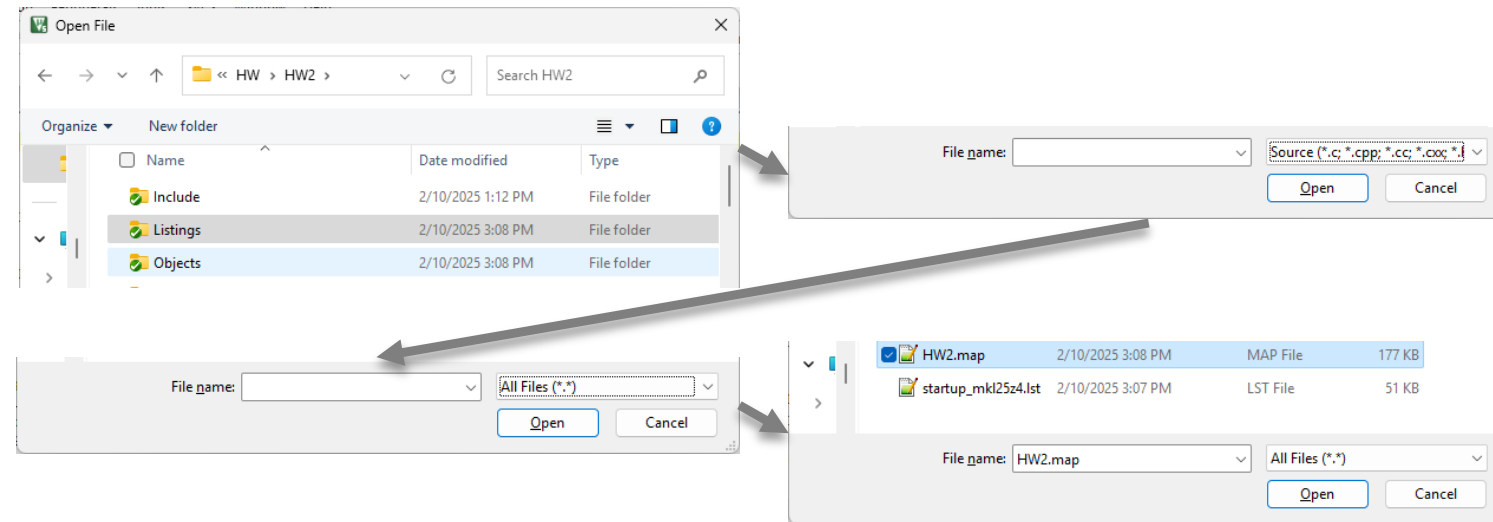
453 }
454
455 /* Write color to the next count pixel locations i
456 You must have called LCD_Start_Rectangle before ca
457 void LCD_Write_Rectangle_Pixel(COLOR_T * color, un
458 uint8_t b1, b2;
459
460 #if DEV_PACKED_PIXEL_COLOR_T
461 b1 = GET_B1(color);
462 b2 = GET_B2(color);
463 #else
464 // 16 bpp, 5-6-5. Assume color channel data is 1
465 b1 = (color->R&0xf0) | ((color->G&0xe0)>>5);
  
```

- Find Start address of region
- Right-click in disassembly window, select "Show Disassembly at Address ..."

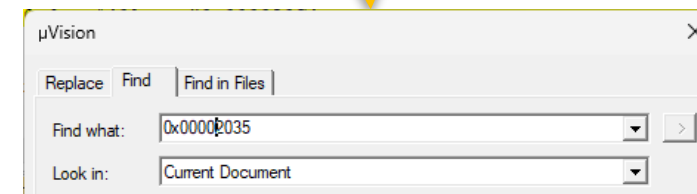
- Type in region's Start address and press "Go To"
- Source code window will show file and code for that address, marked with cyan triangle pointer

## Name Truncation in RegionTable? Option 2

- Select File->Open (Ctrl-O)
  - Enter the Listings folder
  - Change file type from **Source (\*.c; ...)** to **All Files (\*.\*)**
  - Select the project's .map file and click Open
- Identify the region's **Start** address in the RegionTable
- Select the .map file in a debugger source window
- Find (Ctrl-F) that Start address in the map file
- The map file should show the complete function name



RegionTable[SortedRegions[2]]	0x00004CC4
Start	0x00002035
End	0x00002086
Name	0x00004CCC "LCD_Write_Rectangle_Pixe"



The screenshot shows the 'HW2.map' file open in the debugger source window. The file contains a list of functions and their addresses. The function 'LCD\_Write\_Rectangle\_Pixel' is highlighted in green, showing its address as 0x00002035.

LCD_Text_PrintStr	0x00001f59	Thumb Code	152	lcd
LCD_Text_Set_Colors	0x00001ff9	Thumb Code	52	lcd
LCD_Write_Rectangle_Pixel	0x00002035	Thumb Code	82	st7
PIT_IRQHandler	0x00002089	Thumb Code	60	tim
PIT_Init	0x000020c9	Thumb Code	84	tim