

CLK_CPU_vs_CLK_PER.elf: file format elf32-avr

Sections:

Idx	Name	Size	VMA	LMA	File off	Align
0	.data	00000000	00804000	00804000	000001b6	2**0
	CONTENTS, ALLOC, LOAD, DATA					
1	.text	00000162	00000000	00000000	00000054	2**1
	CONTENTS, ALLOC, LOAD, READONLY, CODE					
2	.comment	00000030	00000000	00000000	000001b6	2**0
	CONTENTS, READONLY					
3	.note.gnu.avr.deviceinfo	00000040	00000000	00000000	00000000	000001e8 2**2
	CONTENTS, READONLY					
4	.debug_aranges	00000020	00000000	00000000	00000228	2**0
	CONTENTS, READONLY, DEBUGGING					
5	.debug_info	00003251	00000000	00000000	00000248	2**0
	CONTENTS, READONLY, DEBUGGING					
6	.debug_abbrev	00002e08	00000000	00000000	00003499	2**0
	CONTENTS, READONLY, DEBUGGING					
7	.debug_line	000003bf	00000000	00000000	000062a1	2**0
	CONTENTS, READONLY, DEBUGGING					
8	.debug_frame	00000024	00000000	00000000	00006660	2**2
	CONTENTS, READONLY, DEBUGGING					
9	.debug_str	0000178c	00000000	00000000	00006684	2**0
	CONTENTS, READONLY, DEBUGGING					
10	.debug_loc	00000048	00000000	00000000	00007e10	2**0
	CONTENTS, READONLY, DEBUGGING					
11	.debug_ranges	00000010	00000000	00000000	00007e58	2**0
	CONTENTS, READONLY, DEBUGGING					

Disassembly of section .text:

00000000 <__vectors>:

```

0: 0c 94 7a 00    jmp 0xf4      ; 0xf4 <__ctors_end>
4: 0c 94 84 00    jmp 0x108     ; 0x108 <__bad_interrupt>
8: 0c 94 84 00    jmp 0x108     ; 0x108 <__bad_interrupt>
c: 0c 94 84 00    jmp 0x108     ; 0x108 <__bad_interrupt>
10: 0c 94 84 00    jmp 0x108     ; 0x108 <__bad_interrupt>
14: 0c 94 84 00    jmp 0x108     ; 0x108 <__bad_interrupt>
18: 0c 94 84 00    jmp 0x108     ; 0x108 <__bad_interrupt>
1c: 0c 94 84 00    jmp 0x108     ; 0x108 <__bad_interrupt>
20: 0c 94 84 00    jmp 0x108     ; 0x108 <__bad_interrupt>
24: 0c 94 84 00    jmp 0x108     ; 0x108 <__bad_interrupt>
28: 0c 94 84 00    jmp 0x108     ; 0x108 <__bad_interrupt>
2c: 0c 94 84 00    jmp 0x108     ; 0x108 <__bad_interrupt>
30: 0c 94 84 00    jmp 0x108     ; 0x108 <__bad_interrupt>
34: 0c 94 84 00    jmp 0x108     ; 0x108 <__bad_interrupt>
38: 0c 94 84 00    jmp 0x108     ; 0x108 <__bad_interrupt>
3c: 0c 94 84 00    jmp 0x108     ; 0x108 <__bad_interrupt>
40: 0c 94 84 00    jmp 0x108     ; 0x108 <__bad_interrupt>
44: 0c 94 84 00    jmp 0x108     ; 0x108 <__bad_interrupt>
48: 0c 94 84 00    jmp 0x108     ; 0x108 <__bad_interrupt>

```

```

4c: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
50: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
54: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
58: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
5c: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
60: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
64: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
68: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
6c: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
70: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
74: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
78: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
7c: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
80: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
84: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
88: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
8c: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
90: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
94: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
98: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
9c: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
a0: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
a4: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
a8: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
ac: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
b0: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
b4: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
b8: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
bc: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
c0: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
c4: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
c8: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
cc: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
d0: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
d4: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
d8: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
dc: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
e0: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
e4: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
e8: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
ec: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>
f0: 0c 94 84 00      jmp 0x108      ; 0x108 <__bad_interrupt>

```

00000f4 <__ctors_end>:

```

f4: 11 24          eor r1, r1
f6: 1f be          out 0x3f, r1      ; 63
f8: cf ef          ldi r28, 0xFF        ; 255
fa: cd bf          out 0x3d, r28    ; 61
fc: df e7          ldi r29, 0x7F        ; 127
fe: de bf          out 0x3e, r29    ; 62
100: 0e 94 86 00     call 0x10c          ; 0x10c <main>
104: 0c 94 af 00     jmp 0x15e          ; 0x15e <_exit>

```

```
00000108 <__bad_interrupt>:
```

```
108: 0c 94 00 00    jmp 0    ; 0x0 <__vectors>
```

```
0000010c <main>:
```

```
#include <util/delay.h>
```

```
int main(void)
```

```
{
    PORTA_DIR = PIN7_bm;    //Set PA7 as the output pin for the CLK_PER (CLK_OUT)
10c: 80 e8          ldi r24, 0x80    ; 128
10e: 80 93 00 04    sts 0x0400, r24 ; 0x800400 <__TEXT_REGION_LENGTH__+0x7e0400>
    PORTB_DIR = PIN3_bm;    //Set PB3 as the output
112: 98 e0          ldi r25, 0x08    ; 8
114: 90 93 20 04    sts 0x0420, r25 ; 0x800420 <__TEXT_REGION_LENGTH__+0x7e0420>

    CPU_CCP = CCP_IOREG_gc; //change protected IO register
118: 98 ed          ldi r25, 0xD8    ; 216
11a: 94 bf          out 0x34, r25    ; 52
    //I guess enable the CLK_MAIN prescaler and set prescaler
    //div to whatever value to divide by what value (4) to see
    CLKCTRL.MCLKCTRLB = CLKCTRL_PDIV0_bm | CLKCTRL_PEN_bm;
11c: e0 e6          ldi r30, 0x60    ; 96
11e: f0 e0          ldi r31, 0x00    ; 0
120: 23 e0          ldi r18, 0x03    ; 3
122: 21 83          std Z+1, r18    ; 0x01
    CPU_CCP = CCP_IOREG_gc; //change protected IO register
124: 94 bf          out 0x34, r25    ; 52
    CLKCTRL.MCLKCTRLA = CLKCTRL_CLKOUT_bm; //enable CLKOUT
126: 80 83          st Z, r24

    while (1)
    {
        PORTB_OUT |= PIN3_bm;
128: e4 e2          ldi r30, 0x24    ; 36
12a: f4 e0          ldi r31, 0x04    ; 4
12c: 80 81          ld r24, Z
12e: 88 60          ori r24, 0x08    ; 8
130: 80 83          st Z, r24
        #else
        //round up by default
        __ticks_dc = (uint32_t)(ceil(fabs(__tmp)));
        #endif

        __builtin_avr_delay_cycles(__ticks_dc);
132: 2f e7          ldi r18, 0x7F    ; 127
134: 8a e1          ldi r24, 0x1A    ; 26
136: 96 e0          ldi r25, 0x06    ; 6
138: 21 50          subi r18, 0x01    ; 1
13a: 80 40          sbci r24, 0x00    ; 0
13c: 90 40          sbci r25, 0x00    ; 0
```

```
13e: e1 f7      brne     .-8      ; 0x138 <main+0x2c>
140: 00 c0      rjmp     .+0      ; 0x142 <main+0x36>
142: 00 00      nop
      _delay_ms(500);
      PORTB_OUT &= ~PIN3_bm;
144: 80 81      ld      r24, Z
146: 87 7f      andi     r24, 0xF7 ; 247
148: 80 83      st      Z, r24
14a: 2f e7      ldi     r18, 0x7F ; 127
14c: 8a e1      ldi     r24, 0x1A ; 26
14e: 96 e0      ldi     r25, 0x06 ; 6
150: 21 50      subi     r18, 0x01 ; 1
152: 80 40      sbci     r24, 0x00 ; 0
154: 90 40      sbci     r25, 0x00 ; 0
156: e1 f7      brne     .-8      ; 0x150 <main+0x44>
158: 00 c0      rjmp     .+0      ; 0x15a <main+0x4e>
15a: 00 00      nop
15c: e5 cf      rjmp     .-54     ; 0x128 <main+0x1c>
```

0000015e <_exit>:

```
15e: f8 94      cli
```

00000160 <__stop_program>:

```
160: ff cf      rjmp     .-2      ; 0x160 <__stop_program>
```