Lucerne University of
Applied Sciences and Arts

HOCHSCHULE
LUZERN

Microcontroller

Testatprojekt 1

Horw, 29.11.2009

Projekt	Testatprojekt 1
Dokument	Projektdokumentation
Schule	Hochschule Luzern, Technik & Architektur
Modul	TA.MC.H0901
Projektteam	**Galliker Thomas** Studiengang Informatik (BB) Panorama 6123 Geiss Tel. +41 79 504 80 70 thomas.galliker@stud.hslu.ch **Moser Christoph** Studiengang Informatik (BB) Zugerbergstrasse 41 6314 Unterägeri Tel. +41 79 785 19 07 christoph.moser@stud.hslu.ch
Dozenten	**Prof. Jost Christian**
Letzte Änderung	29. November 2009, 20:09:00 Uhr

Änderungsprotokoll

Version	Datum	Autor	Beschreibung
0.1	19.11.2009	gat	Dokument von Vorlage erstellt, Aufgabenstellung abgeschrieben
0.2	24.11.2009	moc	Jackson-Diagramm eingefügt
0.3	25.11.2009	gat	Abstract
0.4	26.11.2009	gat	Interpretation der Aufgabenstellung
0.5	27.11.2009	gat	Speicherverwaltung dokumentiert
0.6	28.11.2009	moc/gat	Disassemblierung, Realisierung
0.7	29.11.2009	moc/gat	Quellcode eingefügt, Glossar ergänzt, persönliches Fazit
1.0	29.11.2009	moc	Letzte Korrekturen und Abgabe an C.Jost

Inhalt

Abbildungsverzeichnis

Tabellenverzeichnis

Abstract

Im Zuge der Testatübung 8 des Moduls „Microcontroller“ mussten die Studierenden in Zweiergruppen eine Aufgabe zu den Themenbereichen „Chronometry – Input Capture – Logic Analysis“ bewältigen. Dabei ging es insbesondere darum, die Funktionsweise des Timer Moduls TPMx in einer praktischen Arbeit näher kennen zu lernen. Im Zusammenhang mit dieser Aufgabe gab es auch ein erstes Hands-On an Logic Analyzer. Diese wurden zur Messung und Analysierung von generierten Signalen verwendet. Die Aufgabenstellung verlangte von den Studierenden im Wesentlichen die Generierung eines Digitalsignals im vorgegebenen Frequenzbereich (10-1000Hz) sowie mit vorgegebenen Arbeitszyklen (10-90%). Die Vorgehensweise soll nach den Richtlinien des Projektmanagements dokumentiert werden.

1 Ausgangslage

1.1 Aufgabenstellung

Die Zustandswechsel einer vorgegebenen Rechteckfrequenz im Bereich von 10 bis 1000 Hz mit variablem Duty-Cycle im Bereich von 10 bis 90% sollen zeitlich möglichst genau gemessen werden. Dabei sind folgende Vorgaben einzuhalten:

1. Das Timermodul TPM1 erzeugt an einem Ausgangspin des Controllers die zu messende Rechteckfrequenz (Funktionalität gegeben in „generator.c/h“).
2. Das Timermodul TPM2 soll Frequenz und Duty Cycle des ausgegebenen Rechtecksignals möglichst exakt messen. Dafür wird das Signal auf einem anderen Pin des Controllers zurückgeführt.
3. Die zu generierende Frequenz ist in einer 16-Bit Variabel (Wert entspricht der Frequenz in Hz) und der Duty Cycle in einer 8-Bit Variabel (Wert entspricht dem Duty Cycle in %) gespeichert.
4. Optional: Frequenz und Duty Cycle sind durch zwei Taster (z.B. an Port A) wählbar.
5. Die Source Code Files müssen zwingend heissen:
   `main.c/h` Hauptprogramm, allgemeine Definitionen
   `generator.c/h` Realisiert die Rechteckfrequenz
   `meter.c/h` Realisiert die Zeitmessung
6. Die Anzeige der gemessenen Zeit in [us] erfolgt auf 2 Ports (erster Port 10’000er/1’000er, zweiter Port 100er/10er).
7. Die mit TPM2 gemessenen Zeiten sollen mit dem Logic Analyzer LuLa-USB verifiziert werden. Optional: Es können zusätzlich Messungen mit einem KO durchgeführt werden.
8. Die Dokumentation ist gemäss Vorgaben zu erstellen.

1.2 Verwendete Mittel

- Entwicklungsumgebung: Freescale CodeWarrior 5.9.0 (Windows XP)
- Microcontroller: Freescale HCS08
- Logic Analyzer: Janatek Lu-La-Usb (Software Version: 5.41)

1.3 Vorgehensweise

Um die vorgegebene Aufgabenstellung vollumfänglich zu lösen, wird das Gesamtproblem in mehrere Teilprobleme unterteilt. Dieser Prozess wurde teilweise bereits durch den Auftraggeber (Dozent) erledigt. Die Teilprobleme werden als in sich abgeschlossene Probleme betrachtet und einzeln gelöst.

2 Analyse

2.1 Interpretation der Aufgabenstellung

Der Frequenzgenerator wurde als Vorlage zur Verfügung gestellt. Der Generator erzeugt durch die Mitgabe einer Frequenz und eines DutyCycle(%) ein Rechtecksignal. Für die Signalmessung muss ein zweiter Timer erstellt werden, welcher das generierte Signal abtastet. Das heisst das Ausgangsignal vom Frequenzgenerator wird als Eingangssignal für den Timer 2 Channel 0 benötigt. Mithilfe der ermittelten Counter-Werte von Timer 2, soll der Duty Cycle sowie die gemessene Zeit für eine Periode berechnet werden. Der Duty Cycle soll in Prozent auf einem Display des Medusa-Trainers ausgegeben werden. Die gemessene Zeit für eine Periode, soll auf zwei Displays des Medusa-Trainers angezeigt werden, da es sich dabei um eine 16-Bit Zahl handelt.
Die gemessenen Zeiten des selbst erstellten Programms, sollen mithilfe des Logic Analyzer verifiziert werden.

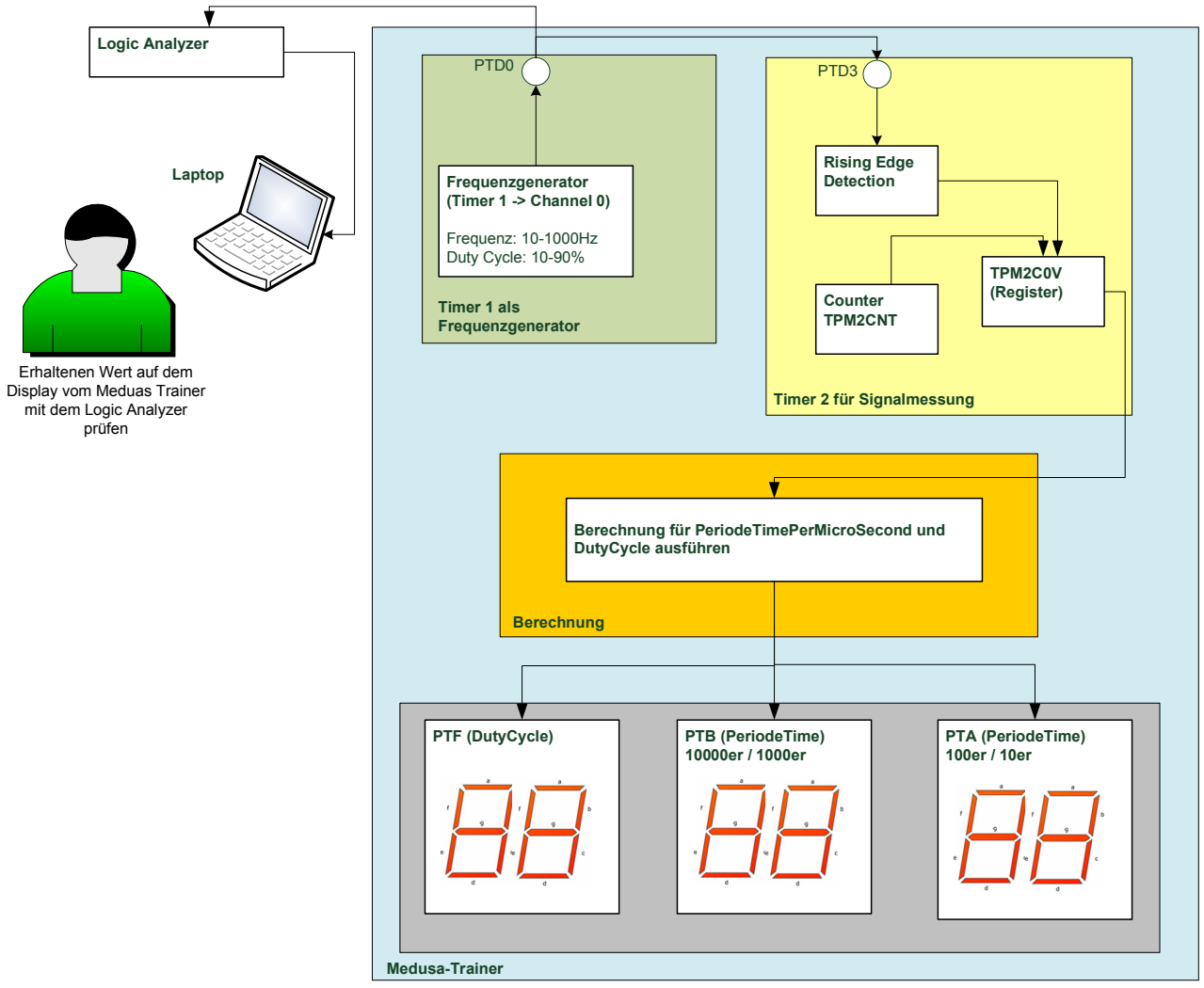

Abbildung 1: Interpretation der Aufgabenstellung

2.2 Systemspezifikation

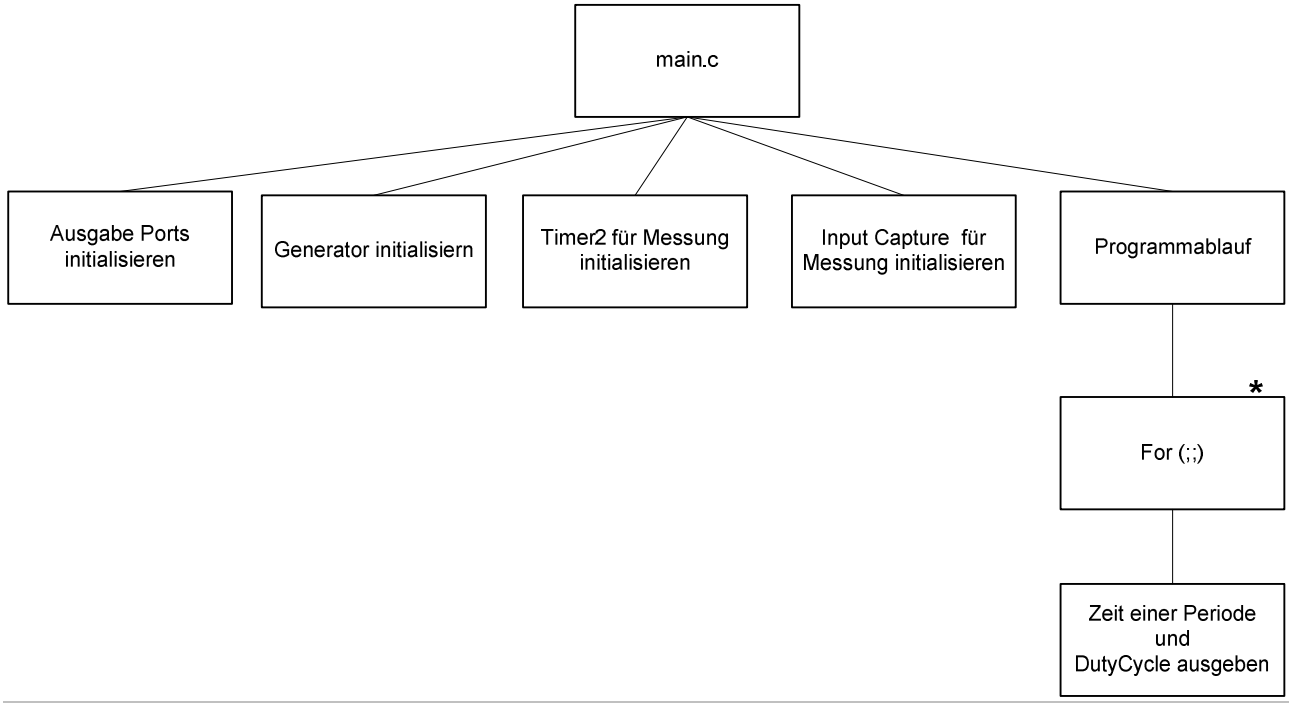

Abbildung 2: Jackson-Baumdiagramm für main.c

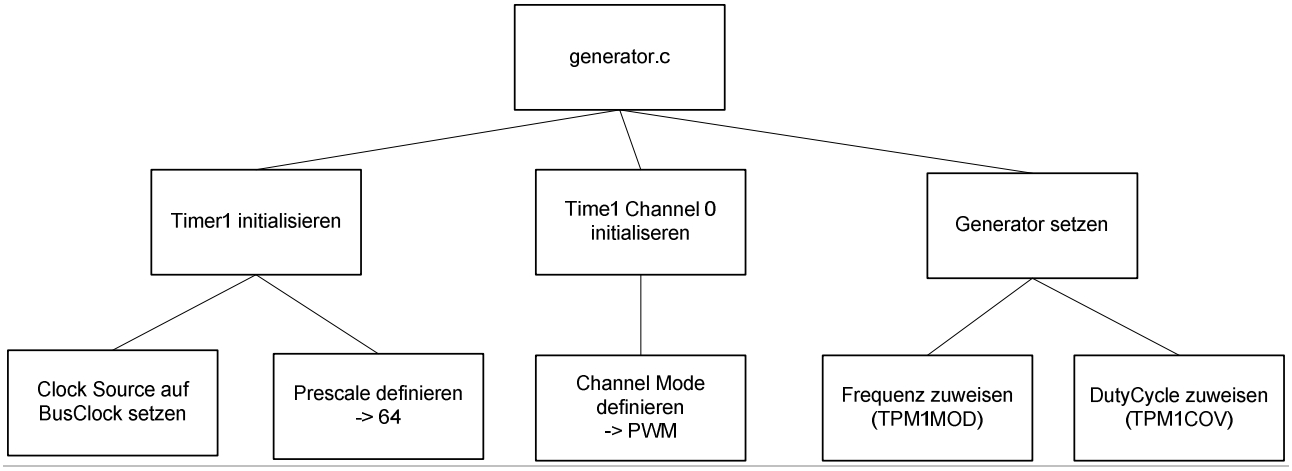

Abbildung 3: Jackson-Baumdiagramm für generator.c

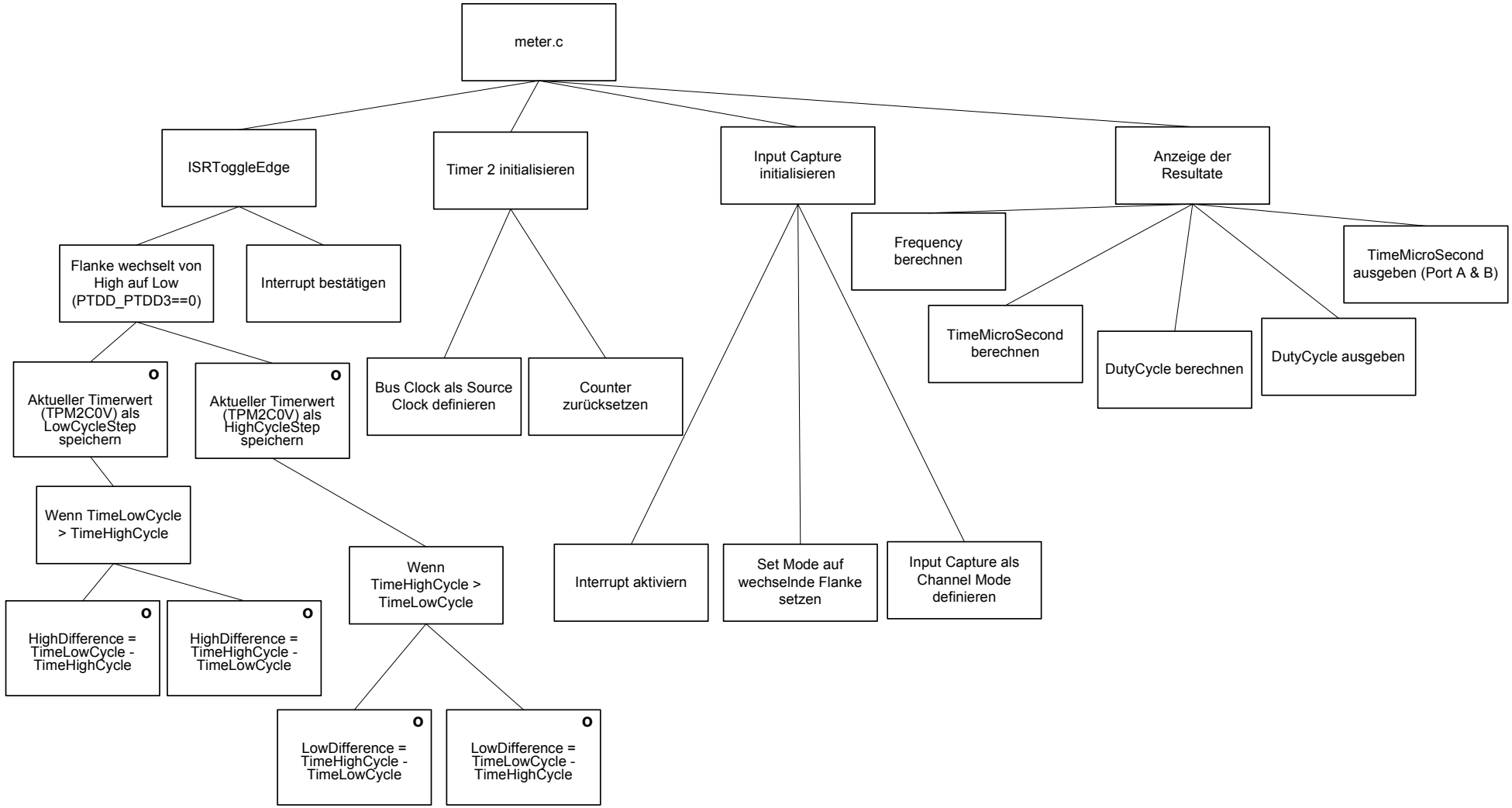

Abbildung 4: Jackson-Baumdiagramm für meter.c

2.3 Anwendungsfälle

Die Anwendungsfälle der Microcontroller Firmware sind auf zwei variable Eingaben beschränkt. Beide Eingaben werden über die Funktion initGenerator eingegeben:

- Variable 1: Die Frequenz (Frequency; in Hz) des zu generierenden Rechtecksignals.
- Variable 2: Der Arbeitszyklus (Duty Cycle; in %) bezogen auf den High Cycle.

Der erlaubte Wertebereich erstreckt sich von 10-1000Hz bzw. 10-90% (in Ganzzahlschritten).

Ein Beispiel: initGenerator(100, 25) generiert ein Rechtecksignal mit einer Frequenz von 100Hz und einem Duty Cycle von 25%. Das ausgegeben Signal entspricht der nachfolgend abgebildeten Visualisierung, wie sie mit einem Logic Analyzer gemacht werden kann:

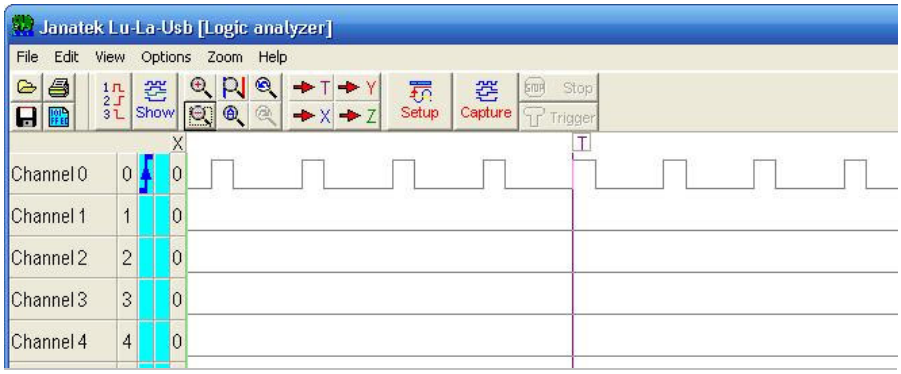

Abbildung 5: Rechtecksignal 25% Duty Cycle, visualisiert im Logic Analyzer.

2.4 Speicherverwaltung

Wie man dem Data Sheet (Kapitel 4) des HCS08 Microcontrollers entnehmen kann, ist das On-Chip Memory in die Bereiche RAM, FLASH, NVRAM sowie I/O-, Kontroll- und Statusregister aufgeteilt. Es gibt folgende Register:

- Direct-page registers (0x0000 bis 0x007F)
- High-page registers (0x1800 bis 0x182B)
- Nonvolatile registers (0xFFB0 bis 0xFFBF)

Startadresse	Endadresse	Beschreibung
0x0000	0x007F	Direct-page registers
0x0080	0x00FF	Z_RAM
0x0100	0x107F	RAM
???		
0x1800	0x182B	High-page registers
0x182C	0xFFAF	ROM
0xFFB0	0xFFBF	Nonvolatile registers
0xFFC0	0xFFCB	ROM2 *(wo wurde ROM1 platziert???)*
0xFFCC	0xFFFF	VECTOR ADDRESS 0xFFCC errISR_RTI // RTI VECTOR ADDRESS 0xFFCE errISR_IIC // IIC VECTOR ADDRESS 0xFFD0 errISR_ATD // ATD conversion VECTOR ADDRESS 0xFFD2 errISR_KBD // Keyboard VECTOR ADDRESS 0xFFD4 errISR_SCI2T // SCI2 transmit VECTOR ADDRESS 0xFFD6 errISR_SCI2R // SCI2 receive VECTOR ADDRESS 0xFFD8 errISR_SCI2E // SCI2 error VECTOR ADDRESS 0xFFDA errISR_SCI1T // SCI1 transmit VECTOR ADDRESS 0xFFDC errISR_SCI1R // SCI1 receive VECTOR ADDRESS 0xFFDE errISR_SCI1E // SCI1 error VECTOR ADDRESS 0xFFE0 errISR_SPI // SPI VECTOR ADDRESS 0xFFE2 errISR_TPM2O // TPM2 overflow VECTOR ADDRESS 0xFFE4 errISR_TPM2CH4 // TPM2 channel 4 VECTOR ADDRESS 0xFFE6 errISR_TPM2CH3 // TPM2 channel 3 VECTOR ADDRESS 0xFFE8 errISR_TPM2CH2 // TPM2 channel 2 VECTOR ADDRESS 0xFFEA errISR_TPM2CH1 // TPM2 channel 1 VECTOR ADDRESS 0xFFEC ISRToggleEdge // TPM2 channel 0 VECTOR ADDRESS 0xFFEE errISR_TPM1O // TPM1 overflow VECTOR ADDRESS 0xFFF0 errISR_TPM1CH2 // TPM1 channel 2 VECTOR ADDRESS 0xFFF2 errISR_TPM1CH1 // TPM1 channel 1 VECTOR ADDRESS 0xFFF4 errISR_TPM1CH0 // TPM1 channel 0 VECTOR ADDRESS 0xFFF6 errISR_ICG // ICG VECTOR ADDRESS 0xFFF8 errISR_LowPower // Low voltage detect VECTOR ADDRESS 0xFFFA errISR_IRQ // IRQ VECTOR ADDRESS 0xFFFC errISR_SWI // SWI VECTOR ADDRESS 0xFFFE _Startup // RESET

Tabelle 1: Tabellarische Ansicht der Speicherverwaltung.

3 Realisierung

Im File meter.c haben wir eine Interrupt ISRToggleEdge erstellt, welcher bei jedem Flankewechsel des eingehenden Signals vom Generator (Timer 1) aufgerufen wird. Im Interrupt werden die folgenden Zeiten ermittelt:

- Periode einer Frequenz High (1)
- Periode einer Frequenz Low (0)

Für die Berechnung und Ausgabe der Werte TimeMicroSecond und DutyCycle wurde eine Methode Show() erstellt, welche im Hauptprogramm in der for-Schleife aufgerufen wird.

3.1 Interrupt Service Routine ISRToggleEdge

Code:

```
/*
* interrupt ISRToggleEdge invokes when the edge status changes
* from low to high as well as from high to low
*/
interrupt void ISRToggleEdge()
{
    // check wether port is 0
    if(PTDD_PTDD3==0)
    {
       // calculate HighDifference (number of clock cycles per high period)
        TimeLowCycle = TPM2C0V;
        if(TimeLowCycle>TimeHighCycle)  // overflow handling
        {
             HighDifference = TimeLowCycle - TimeHighCycle;
        } else
        {
             HighDifference =  TimeHighCycle - TimeLowCycle;
        }
    } else
    {
       // calculate LowDifference (number of clock cycles per low period)
        TimeHighCycle = TPM2C0V;
        if(TimeHighCycle>TimeLowCycle)  // overflow handling
        {
             LowDifference = TimeHighCycle - TimeLowCycle;
        } else
        {
             LowDifference =  TimeLowCycle - TimeHighCycle;
        }
    }
    // acknowledge interrupt
    TPM2C0SC_CH0F = 0;
}
```

Beim Interrupt Aufruf wird mit Hilfe des Wertes an Port D3 (PTDD_PTDD3) geprüft, ob das Signal von High auf Low oder von Low auf High gewechselt hat. Falls das Signal von High auf Low gewechselt hat, wird der aktuelle Counter-Wert vom Timer2 in der Variable TimeLowCycle gespeichert. Danach wird die Zeit für die Periode berechnet in welcher die Flanke den Wert High (1) hatte. Falls das Signal von Low auf High gewechselt hat, wir der aktuelle Counter-Wert vom Timer2 in der Variable TimeHighCycle gespeichert. Danach wird die Zeit für die Periode berechnet in welcher die Flanke den Wert Low (0) hatte. Am Ende des Interrupts wird der Interrupt bestätigt, indem das Interrupt Flag gelöscht wird.

3.2 Funktion show

Code:

```
/*
* Function show calculates frequency, duty cycle as well as frequenc time period
* and puts the values on port A and B
*/
void show(void)
{
    unsigned long freq=0;
    char i=0;
    char c=0;

     // calculate frequency
    freq =  CyclesPerSecond;
    freq /= (HighDifference+LowDifference);

    // calculate time delta of one single frequence period
    TimeMicroSecond = 1000;     // to avoid loss of precision
    TimeMicroSecond *= 1000;    // to avoid loss of precision
    TimeMicroSecond *= (HighDifference+LowDifference);
    TimeMicroSecond /= CyclesPerSecond;

    // calculate Duty Cycle
    DutyCycle = ((HighDifference*100)/(HighDifference+LowDifference));

    PTFD = (char) DutyCycle;


    // output: 10000er and 1000er micro seconds
    i = 16*16*16*16*(TimeMicroSecond / 10000)%10 + 16*16*16*(TimeMicroSecond /
1000)%10;
    PTBD = i;

    // output: 100er and 10er micro seconds
    c = 16*16*(TimeMicroSecond / 100)%10 + 16*(TimeMicroSecond /10)%10;
    PTAD = c;
}
```

Die Show Funktion wird für die Berechnung und Anzeige verwendet. Bei den Berechnungen ist zu beachten, dass diese jeweils in mehreren Schritten gemacht wurden, damit die Präzision nicht verloren geht, da wir die Berechnungen mit Ganzzahlen durchführen.
Bei der Ausgabe auf die Display ist zu beachten, dass diese keine Integer Werte anzeigen können, deshalb wir das Ergebnis vor der Ausgabe mit einem cast in einen char-Wert gewandelt.

3.3 Berechnung TimeMicroSecond

- Bus clock 5Mhz / Prescaler 128 = 39063 cycles per second
- 1 cycle = 1 / 39063 s
- CyclesPerPeriode = (39063/Frequency)
- TimeMicroSecond = (CyclesPerPeriode / 39063) * 1'000'000

3.4 Disassemblierung

Nachfolgend wird die entscheidende Codestelle in meter.c, an welcher unterschieden wird, ob das Eingangssignal von High nach Low oder von Low nach High gewechselt hat, disassembliert. Der Assembler Befehl BRSET prüft, ob ein Bit auf dem Port PTDD_PTDD3 gesetzt ist. Wenn ja, wird abgezweigt.

Code:

```
C Code:           if(PTDD_PTDD3==0)
Assembler Code:   BRSET 3,_PTDD,L3C ;abs = 003c
```

4 Testkonzept

Aus den beschriebenen Anwendungsfällen resultieren grundsätzlich die möglichen Testfälle. Die Funktion initGenerator nimmt zwei Argumente entgegen. Dabei wissen wir, dass die eingegebene Frequenz zwischen 10-1000Hz und der Duty Cycle zwischen 10-90% sein darf. Theoretisch würde diese Kombination genau (990*80=) 79'200 Testfälle geben. Um nicht alle Fälle testen zu müssen, werden wir uns auf folgende Auswahl beschränken:

- 3 zulässige Testfälle
- 3 zulässige Spezialfälle
- 3 nicht zulässige Testfälle

4.1 Testfälle und -abläufe

Testfälle sollten mit Anwendungsfällen mappen.

	Testfall	**IST-Wert** (Δt in µs / Duty Cycle in %)	**SOLL-Wert*** (Δt in µs / Duty Cycle in %)	**Zulässigkeit**
Zulässige Testfälle				
1	initGenerator(200, 30);	4991 / 29	5000 / 30	**JA**
2	initGenerator(500, 50);	1996 / 48	2000 / 50	**JA**
3	initGenerator(800, 75);	1254 / 75	1250 / 75	**JA**
Zulässige Spezialfälle				
1	initGenerator(10, 10);	99966 / 9	100000 / 10	**JA**
2	initGenerator(1000, 10);	998 / 8	1000 / 10	**JA**
3	initGenerator(1000, 90);	998 / 87	1000 / 90	**JA**
Nicht zulässige Testfälle				
1	initGenerator(0, 0);	0 / 0	DivByZero / 0	**NEIN**
2	initGenerator(0, 50);	0 / 0	DivByZero / 0	**NEIN**
3	initGenerator(500, 100);	54973 / 43	maxTime / 100	**NEIN**

Tabelle 2: Auflistung der ausgeführten Testfälle.

* anhand der folgenden Formel überprüft: TimeMicroSecond = (CyclesPerPeriode / 39063) * 1000000
CyclesPerPeriode = (39063/Frequenz)

5 Persönliches Fazit

5.1 Fazit von Christoph Moser

→ In Aufgaben mit dem Mikrocontroller ist es wichtig das Datenblatt(Datensheet) genau zu verstehen. Beispielsweise hatten wir viel Zeit benötigt, bis wir herausgefunden hatten, dass das Input Signal nicht an den von definierten Channel gesendet wurde, da wir den falschen Pin verkabelt hatten.
→ Programmieraufgaben benötigen meistens mehr Zeit als ursprünglich eingeplant, da es immer zu unerwarteten Problemen kommt.
→ Hands-on Workshops sind sehr lehrreich und das gelernte bleibt besser hängen.

5.2 Fazit von Thomas Galliker

→ Hands-on Workshops sind viel lehrreicher als Theorielektionen.
→ Anspruchsvolle aber lehrreiche Aufgabe, insbesondere für Informatikstudierende.
→ Viel zu wenig Zeit für ein derart aufwändiges Testat.
→ Ungenügende Debug-Möglichkeiten mit der CodeWarrior IDE.

6 Anhang

6.1 Bilder

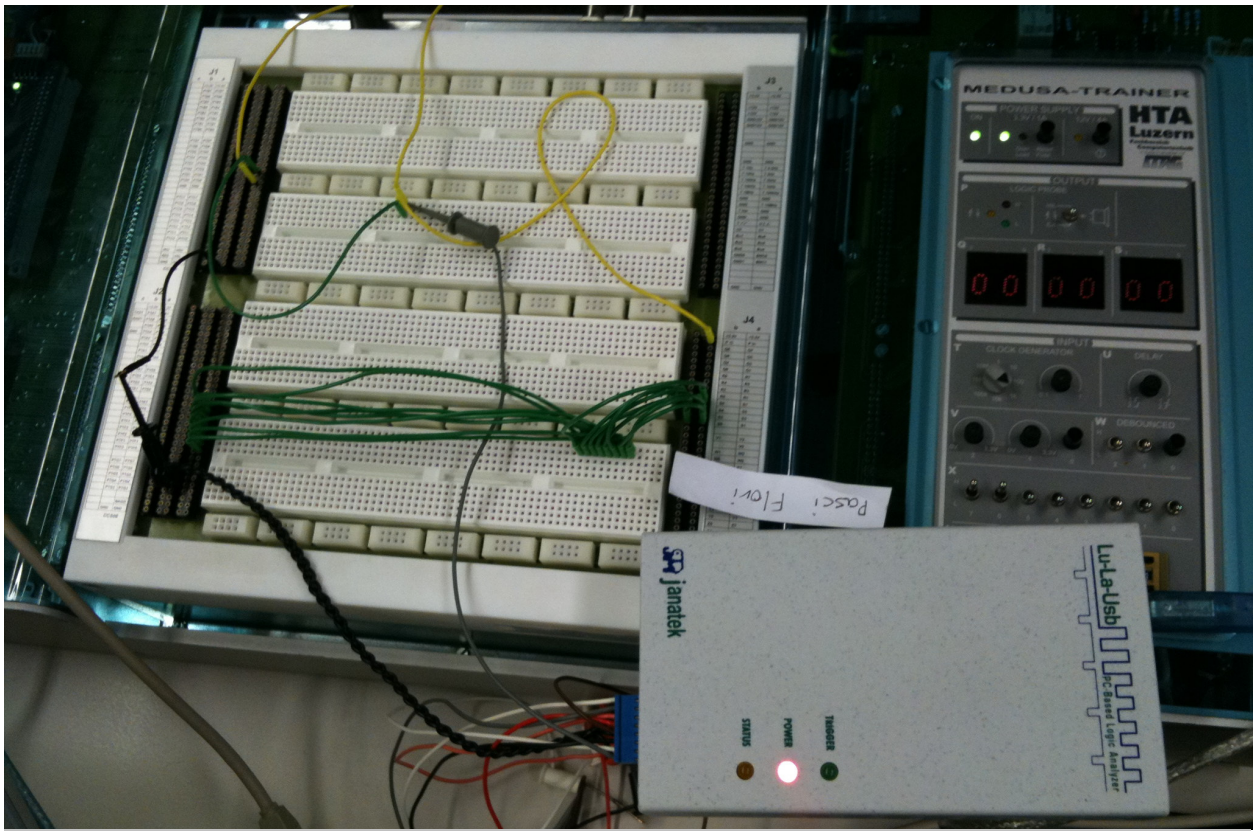

Abbildung 6: Das Ausgangssignal wurde mit dem Logic Analyzer abgegriffen und geprüft

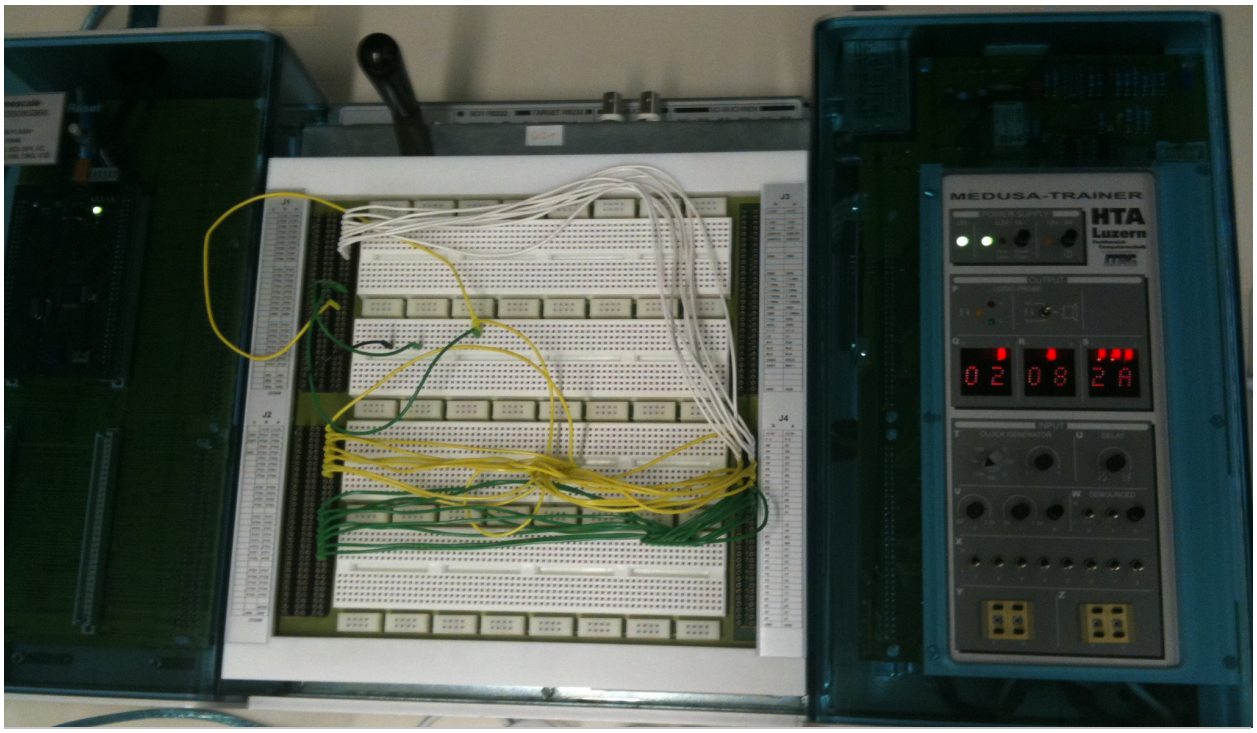

Abbildung 7: Kompletter Versuchsaufba mit Medusa Box und Medusa Trainer

6.2 Glossar

Branch
Bedeutet eine Abzweigung beispielsweise in der Krone eines Baums. Der Ausdruck wird in der Assemblerprogrammierung zur Signalisierung einer Programmverzweigung verwendet. Beispiele dafür sind: BRN=Branch Never, BRCLR=Branch if bit cleared, BRSET=Branch if bit set.

CPU Cycle
Ein einzelner Takt, welcher von einer CPU ausgelöst wird. Eine 20MHz CPU generiert 20'000'000 CPU Cycles pro Sekunde.

Prescaler
Ein Prescaler ist ein elektronischer Zähler, welcher eine hohe Frequenz durch Ganzzahlteilung in eine kleinere Frequenz teilt. Beispiel: Eine 20MHz Frequenz wird mit einem Prescaler von 4 geteilt, so entsteht eine Ausgangsfrequenz von 5MHz.

PWM
Pulse-width modulation
Die Pulsweitenmodulation ist eine Modulationsart, bei der eine technische Grösse (z.B. elektrischer Strom) zwischen zwei Werten wechselt. Dabei wird bei konstanter Frequenz das Tastverhältnis des Signals moduliert, also die Breite eines Impulses.

6.3 Quellcode

6.3.1 derivative.h

```
/* Include the derivative-specific header file */
#include <MC9S08GB60A.h>

#define _Stop asm ( stop; )
  /*!< Macro to enter stop modes, STOPE bit in SOPT1 register must be set prior to
executing this macro */

#define _Wait asm ( wait; )
  /*!< Macro to enter wait mode */
```

6.3.2 generator.c

```
/**
 *--------------------------------------------------------------------\n
 *          HSLU T&A Hochschule Luzern Technik+Architektur             \n
 *--------------------------------------------------------------------\n
 *
 * \brief         HCS08 Zeitmesser - Test&Measurement, Uebung8
 * \file
 * \author        Zabkar Daniel, daniel.zabkar@hslu.ch
 * \date          06.11.2009
 *
 * Language:   Ansi-C \n\n
 * Target:     Medusa-Box / Demoboard mit HCS08 \n
 *
 *
 *  $Id: main.c 134 2009-09-09 11:49:49Z zajost
 *
 * \par Description:
 * HCS08 TPM 1 Initialisierung und Funktionsgenerator, Uebung8
 *
 * \par History:
 *   1.0      06.11.09   Zad    Erstellung \n
 *--------------------------------------------------------------------
 */

#include "generator.h"

/**
 * Deaktiviert den Watchdog und konfiguriert den Takt.
 *
 * Konfiguration des ICG (Internal Clock Generator):
 * FEI => FLL Engaged, Internal Clock (FEI) Mode.
 *
 * FEI bedeutet, dass kein externer Clock verwendet wird,
 * sondern der interne 243 kHz Referenzclock. Diese Frequenz
 * wird mit Hilfe des FLL (Frequency Locked Loop) hoch
 * multipliziert, damit sich schlussendlich eine Busfrequenz
 * von ca. 5 MHz einstellt.
 *
 * Über das ICGTRM-Register könnten die 5MHz zusätzlich abgestimmt
 * werden.
 */

void initClock(void)
{
    SOPT    = 0x73;     // Watchdog=disabled, Stop-Mode=enabled

    ICGC1   = 0x28;     // ICG-Mode: FEI
    ICGC2   = 0x72;     // N=10, R04
    while(!ICGS1_LOCK); // Warten bis Clock stabil (locked) ist.
}
```

```
/**
 * Initialisierung des TPM1-Moduls und starten des Generators
 * Diese Funktion muss im main.c aufgerufen werden um den
 * Generator zu starten!
 *
 * @param [in] _uiFreq      : GenFrequenz [Hz]
 * @param [in] _cDC         : DutyCycle [%]
 *
 */

void initGenerator(unsigned int uiFreq, unsigned char ucDC )
{
    /* TPM1 initialization */
    /* Set TPM Status and Control Registers:*/
    (void)TPM1SC;   // Read register so flag is cleared
    TPM1SC      =   ( BUS_CLOCK | DIVIDE_BY_64 );

    /* TPM1 Channel 0 initialization*/
    /* Set TPM1 Channel 0 Status and Control Registers:*/
    (void)TPM1C0SC; // Read register so flag is cleared
    TPM1C0SC    =   (EDGE__PWM_CLEAR);

    setGenerator(uiFreq, ucDC);
}


/**
 * Setzt die aktuelle Frequenz und DutyCycle des Generators
 *
 * @param [in] _uiFreq      : GenFrequenz [Hz]
 * @param [in] _cDC         : DutyCycle [%]
 *
 */

void setGenerator(unsigned int uiFreq, unsigned char ucDC )
{
    // Setzt die aktuelle Frequenz des Generators im Timer 1
    //Modulo Register
    TPM1MOD     =   (unsigned int)(FUNC_FREQ/uiFreq-1);
    // Setzt den aktuellen DutyCycle des Generators im Timer 1
    // Channel 0 Modulo Register
    TPM1C0V     =   (unsigned int)(((FUNC_FREQ/uiFreq-1)*ucDC)/100);
}


interrupt void ISR_TPM1CH0(void) {
      TPM1C0SC_CH0F = 0;
}
```

6.3.3 generator.h

```
/**
 *--------------------------------------------------------------------\n
 *          HSLU T&A Hochschule Luzern Technik+Architektur             \n
 *--------------------------------------------------------------------\n
 *
 * \brief         HCS08 Zeitmesser - Test&Measurement, Uebung8
 * \file
 * \author        Zabkar Daniel, daniel.zabkar@hslu.ch
 * \date          05.11.2009
 *
 * Language:   Ansi-C \n\n
 * Target:     Medusa-Box / Demoboard mit HCS08 \n
 *
 *  $Id: main.c 134 2009-09-09 11:49:49Z zajost
 *
 * \par Description:
 * HCS08 TPM 1 Initialisierung und Funktionsgenerator, Uebung8
 *
 * \par History:
 *   1.0    Erstellung Zad
 *--------------------------------------------------------------------
 */

#ifndef GENERATOR_H
#define GENERATOR_H

#include <hidef.h>      // for EnableInterrupts macro
#include "derivative.h" // include peripheral declarations

//Timer Selects

//TOF interrupts enabled (Bit 6)
#define TOIE                (0x40)

//All TPMx channels operate in center-aligned PWM mode. (Bit 5)
#define CPWMS               (0x20)


//Clock Source Select (Bit 3-4)
#define BUS_CLOCK           (0x08)
#define FIXED_CLOCK         (0x10)
#define EXTERNAL_OSC        (0x18)

//Prescale Divisor Select (Bit 0-2)
#define DIVIDE_BY_1         (0x00)
#define DIVIDE_BY_2         (0x01)
#define DIVIDE_BY_4         (0x02)
#define DIVIDE_BY_8         (0x03)
#define DIVIDE_BY_16        (0x04)
#define DIVIDE_BY_32        (0x05)
#define DIVIDE_BY_64        (0x06)
#define DIVIDE_BY_128       (0x07)

//Channel Selects

//Channel 0 interrupt requests enabled (Bit 6)
#define CHxIE               (0x40)

//Mode, Edge, and Level Selection (Bit 4-5)
#define INPUT_CAPTURE       (0x00)
#define OUTPUT_COMPARE      (0x10)

//Mode, Edge, and Level Selection / Configuration (Bit 2-3)
#define CAPTURE_RISING      (0x04)
#define CAPTURE_FALLING     (0x08)
```

```
#define CAPTURE_BOTH        (0x0C)

#define OUTPUT_TOGGLE       (0x04)
#define OUTPUT_CLEAR        (0x08)
#define OUTPUT_SET          (0x0C)

//the counter overflow forces the PWM signal high and the output
//compare forces the PWM signal low
#define EDGE__PWM_CLEAR     (0x38)

//the counter overflow forces thePWMsignal low and the output
//compare forces the PWM signal high
#define EDGE__PWM_SET       (0x3C)

#define disableTPM_ISR()  (TPM1C0SC &= ~(CHxIE))
#define enableTPM_ISR()   (TPM1C0SC |=  (CHxIE))

//Frequenzen

// The bus clock frequency is 4.99890 MHz
#define OSC_FREQ            4998900

// OSC_FREQ / 64
#define FUNC_FREQ           78108

/************************ Prototypes ****************************************/

void initClock(void);
void initGenerator (unsigned int, unsigned char);
void setGenerator(unsigned int, unsigned char);

#endif
```

6.3.4 main.c

```
/**
 *---------------------------------------------------------------------\n
 *          HSLU T&A Hochschule Luzern Technik+Architektur             \n
 *---------------------------------------------------------------------\n
 *
 *brief         HCS08 Zeitmesser - Test&Measurement, Uebung8
 *file
 *author        Christoph Moser, Thomas Galliker
 *date          19.11.2009
 *
 *Language:     Ansi-C
 *Target:       Medusa-Box
 *
 *Description:
 *measure the edge status change
 *
 *History:
 *  21.11.09 MoC/GaT: create initPorts and main function
 *---------------------------------------------------------------------
 */
#include <hidef.h>      // for EnableInterrupts macro
#include "derivative.h" // include peripheral declarations
#include "generator.h"  // include generator
#include "meter.h"      // include meter

/**
 * Initialise Output Ports
 * - Port F -> DutyCycle
 * - Port C -> Frequenz 10000er / 1000er
 * - Port A -> Frequenz 100er / 10er
 */
void initPorts(void)
{
    PTFDD = 0xff;           // Port F = Output -> DutyCycle
    PTBDD = 0xff;           // Port C = Output -> Frequency  10000er/1000er
    PTADD = 0xff;           // Port A = Output -> Frequency  100er/10er

    PTFD = 0x0;
    PTBD = 0x0;
    PTAD = 0x0;
}
/**
 * Main
 * initialise the Generator and the Timer and Input Capture for the measurement
 */
void main(void)
{
    initClock();            // configure Clock and Watchdog

    initPorts();            // configure Ports

    EnableInterrupts;       // activate Interrupts

    initGenerator(500, 100); // initialise generator -> (frequency, dutycycle)

    initTimer();             // initialise timer2

    initInputCapture();      // initialise the channel0 on timer2

    for(;;)
    {
        show();
    }
}
```

6.3.5 meter.c

```
/**
 *---------------------------------------------------------------------\n
 *          HSLU T&A Hochschule Luzern Technik+Architektur             \n
 *---------------------------------------------------------------------\n
 *
 *brief         HCS08 Zeitmesser - Test&Measurement, Uebung8
 *file
 *author        Christoph Moser, Thomas Galliker
 *date          19.11.2009
 *
 *Language:     Ansi-C
 *Target:       Medusa-Box
 *
 *Description:
 *calculates the frequenz and DutyCycle
 *
 *History:
 *21.11.09 MoC/GaT: create initTimer, InputCapture and the Interrupt
 */

#include "derivative.h" // include peripheral declarations

static unsigned long CyclesPerSecond = 39063;
static unsigned long TimeHighCycle=0;
static unsigned long TimeLowCycle=0;
static unsigned long HighDifference=0;
static unsigned long LowDifference=0;
static unsigned long TimeMicroSecond = 0;
static unsigned long DutyCycle = 0;

/**
 * initialise the TPM2-Module
 */
void initTimer(void)
{
    //define bus clock
    TPM2SC_CLKSB = 0;
    TPM2SC_CLKSA = 1;

    // set prescaler to 128 (5 Mhz / 128 -> 39063 cycles/sec)
    TPM2SC_PS0 = 1;
    TPM2SC_PS1 = 1;
    TPM2SC_PS2 = 1;

    // set modulo
    TPM2MOD = 0;

    // main counter auf 0 zurücksetzen
    TPM2CNT = 0;
}
/**
* initialise channel0 of the TPM2-Module as input capture
*/
void initInputCapture(void)
{
    // enable Interrupt
    TPM2C0SC_CH0IE = 1;

    // set channel mode for Input Capture
    TPM2C0SC_MS0B = 0;
    TPM2C0SC_MS0A = 0;

    // set mode -> Capture on risinig or falling edge
    TPM2C0SC_ELS0B = 1;
    TPM2C0SC_ELS0A = 1;
}
```

```
/*
* interrupt ISRToggleEdge invokes when the edge status changes
* from low to high as well as from high to low
*/
interrupt void ISRToggleEdge()
{
    // check wether port is 0
    if(PTDD_PTDD3==0)
    {
       // calculate HighDifference (number of clock cycles per high period)
        TimeLowCycle = TPM2C0V;
        if(TimeLowCycle>TimeHighCycle)  // overflow handling
        {
             HighDifference = TimeLowCycle - TimeHighCycle;
        } else
        {
             HighDifference =  TimeHighCycle - TimeLowCycle;
        }
    } else
    {
             // calculate LowDifference (number of clock cycles per low period)
        TimeHighCycle = TPM2C0V;
        if(TimeHighCycle>TimeLowCycle)  // overflow handling
        {
             LowDifference = TimeHighCycle - TimeLowCycle;
        } else
        {
             LowDifference =  TimeLowCycle - TimeHighCycle;
        }
    }
    // acknowledge interrupt
    TPM2C0SC_CH0F = 0;
}
/*
* Function show calculates frequency, duty cycle as well as frequenc time period
* and puts the values on port A and B
*/
void show(void)
{
    unsigned long freq=0;
    char i=0;
    char c=0;

     // calculate frequency
    freq =  CyclesPerSecond;
    freq /= (HighDifference+LowDifference);

    // calculate time delta of one single frequence period
    TimeMicroSecond = 1000;             // to avoid loss of precision
    TimeMicroSecond *= 1000;    // to avoid loss of precision
    TimeMicroSecond *= (HighDifference+LowDifference);
    TimeMicroSecond /= CyclesPerSecond;

    // calculate Duty Cycle
    DutyCycle = ((HighDifference*100)/(HighDifference+LowDifference));
    PTFD = (char) DutyCycle;

    // output: 10000er and 1000er micro seconds
    i = 16*16*16*16*(TimeMicroSecond / 10000)%10 + 16*16*16*(TimeMicroSecond / 1000)%10;
    PTBD = i;

    // output: 100er and 10er micro seconds
    c = 16*16*(TimeMicroSecond / 100)%10 + 16*(TimeMicroSecond /10)%10;
    PTAD = c;
}
```

6.3.6 meter.h

```
/*-------------------------------------------------------------------\n
 *          HSLU T&A Hochschule Luzern Technik+Architektur           \n
 *-------------------------------------------------------------------\n
 *
 *brief         HCS08 Zeitmesser - Test&Measurement, Uebung8
 *file
 *author        Christoph Moser, Thomas Galliker
 *date          19.11.2009
 *
 *Language:     Ansi-C
 *Target:       Medusa-Box
 *
 *
 *Description:
 *calculates the frequenz and DutyCycle
 *
 *History:
 * 21.11.09 MoC/GaT: create initTimer, InputCapture and the Interrupt
 */
 void initTimer(void);
 void initInputCapture(void);
 void show(void);
```

6.3.7 Start08.c

```
/****************************************************************************
  FILE        : start08.c
  PURPOSE     : 68HC08 standard startup code
  LANGUAGE    : ANSI-C / INLINE ASSEMBLER
  ----------------------------------------------------------------------------
  HISTORY
    22 oct 93         Created.
    04/17/97          Also C++ constructors called in Init().
 ****************************************************************************/

/*********************************************************************/
/* NOTE:                                                             */
/* This version of the startup code assumes that main                */
/* does never return (saving the 2 byte return address of _Startup on */
/* the stack).                                                       */
/*********************************************************************/

#define __NO_FLAGS_OFFSET   /* we do not need the flags field in the startup
data descriptor */
#define __NO_MAIN_OFFSET    /* we do not need the main field in the startup data
descriptor */

#include <start08.h>

#ifdef __cplusplus
#define __EXTERN_C  extern "C"
#else
#define __EXTERN_C
#endif

__EXTERN_C extern void main(void); /* prototype of main function */

#include "non_bank.sgm"


/*********************************************************************/
/* Macros to control how the startup code handles the COP:           */
```

```
/* #define _DO_FEED_COP_   : do feed the COP                                */
/* Without defining any of these, the startup code does NOT handle the COP */
/***************************************************************************/
/* __ONLY_INIT_SP define:                                                  */
/* This define selects an shorter version of the startup code              */
/* which only loads the stack pointer and directly afterwards calls        */
/* main. This version does however NOT initialize global variables         */
/* (so this version is not ANSI compliant!).                               */
/***************************************************************************/

#if defined(_DO_FEED_COP_)
#define __FEED_COP_IN_HLI()  } _FEED_COP(); __asm {
#else
#define __FEED_COP_IN_HLI() /* do nothing */
#endif

#ifndef __ONLY_INIT_SP

#pragma DATA_SEG FAR _STARTUP
struct _tagStartup _startupData;    /* read-only:
                                       _startupData is allocated in ROM and
                                       initialized by the linker */

#pragma MESSAGE DISABLE C20001 /* Warning C20001: Different value of
stackpointer depending on control-flow */
/* the function _COPY_L releases some bytes from the stack internally */

#if defined(__OPTIMIZE_FOR_SIZE__) || defined(_DO_FEED_COP_)
#pragma NO_ENTRY
#pragma NO_EXIT
#pragma NO_FRAME
/*lint -esym(528, loadByte) inhibit warning about unreferenced loadByte function
*/
static void near loadByte(void) {
  asm {
              PSHH
              PSHX
#ifdef __HCS08__
              LDHX    5,SP
              LDA     0,X
              AIX     #1
              STHX    5,SP
#else
              LDA     5,SP
              PSHA
              LDX     7,SP
              PULH
              LDA     0,X
              AIX     #1
              STX     6,SP
              PSHH
              PULX
              STX     5,SP
#endif
              PULX
              PULH
              RTS
  }
}
#endif /* defined(__OPTIMIZE_FOR_SIZE__) || defined(_DO_FEED_COP_) */
```

```
#ifdef __cplusplus
static void Call_Constructors(void) {
  int i;
#ifdef __ELF_OBJECT_FILE_FORMAT__
  i = (int)(_startupData.nofInitBodies - 1);
  while (i >= 0) {
    (&_startupData.initBodies->initFunc)[i]();  /* call C++ constructors */
    i--;
  }
#else /* __ELF_OBJECT_FILE_FORMAT__ */
  /* HIWARE object file format */
  if (_startupData.mInits != NULL) {
    _PFunc *fktPtr;
    fktPtr = _startupData.mInits;
    while(*fktPtr != NULL) {
      (**fktPtr)(); /* call constructor */
      fktPtr++;
    }
  }
#endif /* __ELF_OBJECT_FILE_FORMAT__ */
}
#endif

/*lint -esym(752,_COPY_L)  inhibit message on function declared, but not used
(it is used in HLI) */
__EXTERN_C extern void _COPY_L(void);
/* DESC:    copy very large structures (>= 256 bytes) in 16-bit address space
(stack incl.)
   IN:      TOS count, TOS(2) @dest, H:X @src
   OUT:
   WRITTEN: X,H */
#ifdef __ELF_OBJECT_FILE_FORMAT__
      #define toCopyDownBegOffs 0
#else
      #define toCopyDownBegOffs 2 /* for the hiware format, the toCopyDownBeg
field is a long. Because the HC08 is big endian, we have to use an offset of 2
*/
#endif
static void Init(void) {
/* purpose:     1) zero out RAM-areas where data is allocated
                2) init run-time data
                3) copy initialization data from ROM to RAM
 */
  /*lint -esym(529,p,i)  inhibit warning about symbols not used: it is used in
HLI below */
  int i;
  int *far p;

  asm {
ZeroOut:
             LDA    _startupData.nofZeroOuts:1 ; // nofZeroOuts
             INCA
             STA    i:1                        ; // i is counter for number of
zero outs
             LDA    _startupData.nofZeroOuts:0 ; // nofZeroOuts
             INCA
             STA    i:0
             LDHX   _startupData.pZeroOut      ; // *pZeroOut
             BRA    Zero_5
Zero_3:
```

```
              ; // CLR    i:1 is already 0
Zero_4:
              ; // { HX == _pZeroOut }
              PSHX
              PSHH
              ; // { nof bytes in (int)2,X }
              ; // { address in (int)0,X   }
              LDA    0,X
              PSHA
              LDA    2,X
              INCA
              STA    p                 ; // p:0 is used for high byte of byte
counter
              LDA    3,X
              LDX    1,X
              PULH
              INCA
              BRA    Zero_0
Zero_1:
            ;  // CLRA   A is already 0, so we do not have to clear it
Zero_2:
              CLR    0,X
              AIX    #1
              __FEED_COP_IN_HLI()          ; // it's necessary to feed the COP in
the inner loop for the fast COP timeout of some derivatives
Zero_0:
              DBNZA  Zero_2
Zero_6:
              DBNZ   p, Zero_1
              PULH
              PULX                        ; // restore *pZeroOut
              AIX    #4                   ; // advance *pZeroOut
Zero_5:
              DBNZ   i:1, Zero_4
              DBNZ   i:0, Zero_3

CopyDown:

  }

  /* copy down */
  /* _startupData.toCopyDownBeg  --->  {nof(16) dstAddr(16) {bytes(8)}^nof}
Zero(16) */
#if defined(__OPTIMIZE_FOR_SIZE__) || defined(_DO_FEED_COP_) /* for now: only -
os version supports _DO_FEED_COP_ */
  asm {
#ifdef __HCS08__
              LDHX   _startupData.toCopyDownBeg:toCopyDownBegOffs
              PSHX
              PSHH
#else
              LDA    _startupData.toCopyDownBeg:(1+toCopyDownBegOffs)
              PSHA
              LDA    _startupData.toCopyDownBeg:(0+toCopyDownBegOffs)
              PSHA
#endif
Loop0:
              JSR    loadByte           ; // load high byte counter
              TAX                       ; // save for compare
              INCA
              STA    i
```

```
              JSR    loadByte          ; // load low byte counter
              INCA
              STA    i:1
              DECA
              BNE    notfinished
              CBEQX  #0, finished
notfinished:

              JSR    loadByte          ; // load high byte ptr
              PSHA
              PULH
              JSR    loadByte          ; // load low byte ptr
              TAX                      ; // HX is now destination pointer
              BRA    Loop1
Loop3:
Loop2:
              __FEED_COP_IN_HLI()
              JSR    loadByte          ; // load data byte
              STA    0,X
              AIX    #1
Loop1:
              DBNZ   i:1, Loop2
              DBNZ   i:0, Loop3
              BRA    Loop0

finished:
              AIS #2
    }
#else /*defined(__OPTIMIZE_FOR_SIZE__) || defined(_DO_FEED_COP_) */
  /* time optimized asm version. */
  asm {
#ifdef __HCS08__
              LDHX   _startupData.toCopyDownBeg:toCopyDownBegOffs
#else
              LDX    _startupData.toCopyDownBeg:(0+toCopyDownBegOffs)
              PSHX
              PULH
              LDX    _startupData.toCopyDownBeg:(1+toCopyDownBegOffs)
#endif
next:
              LDA   0,X                ; // list is terminated by 2 zero bytes
              ORA   1,X
              BEQ   copydone
              PSHX                     ; // store current position
              PSHH
              LDA   3,X                ; // psh dest low
              PSHA
              LDA   2,X                ; // psh dest high
              PSHA
              LDA   1,X                ; // psh cnt low
              PSHA
              LDA   0,X                ; // psh cnt high
              PSHA
              AIX   #4
              JSR   _COPY_L            ; // copy one block
              PULH
              PULX
              TXA
              ADD   1,X                ; // add low
              PSHA
              PSHH
```

```
            PULA
            ADC   0,X               ; // add high
            PSHA
            PULH
            PULX
            AIX   #4
            BRA next
copydone:
  }
#endif /* defined(__OPTIMIZE_FOR_SIZE__) || defined(_DO_FEED_COP_) */

  /* FuncInits: for C++, this are the global constructors */
#ifdef __cplusplus
  Call_Constructors();
#endif /* __cplusplus */

  /* implement ROM libraries initialization here (see startup.c) */
}
#endif /* __ONLY_INIT_SP */


#pragma NO_EXIT
__EXTERN_C void _Startup(void) {
/* set the reset vector to _Startup in the linker parameter file (*.prm):
    'VECTOR 0 _Startup'

    purpose:    1)  initialize the stack
                2)  initialize run-time, ...
                    initialize the RAM, copy down init data, etc (Init)
                3)  call main;
    called from: _PRESTART-code generated by the Linker
*/
  INIT_SP_FROM_STARTUP_DESC();

#ifndef  __ONLY_INIT_SP
  Init();
#endif
#ifndef __BANKED__
  __asm JMP main; /* with a C style main(); we would push the return address on
the stack wasting 2 RAM bytes */
#else
  __asm CALL main;
#endif
}

// ACHTUNG
// Es wurde ein Interrupt-Vektor angesprungen für den
// keine Interrupt-Service-Routine existiert.
interrupt void errISR_RTI() { for (;;)      asm BGND; }
interrupt void errISR_IIC() { for (;;)      asm BGND; }
interrupt void errISR_ATD() { for (;;)      asm BGND; }
interrupt void errISR_KBD() { for (;;)      asm BGND; }
interrupt void errISR_SCI2T() { for (;;)    asm BGND; }
interrupt void errISR_SCI2R() { for (;;)    asm BGND; }
interrupt void errISR_SCI2E() { for (;;)    asm BGND; }
interrupt void errISR_SCI1T() { for (;;)    asm BGND; }
interrupt void errISR_SCI1R() { for (;;)    asm BGND; }
interrupt void errISR_SCI1E() { for (;;)    asm BGND; }
interrupt void errISR_SPI() { for (;;)      asm BGND; }
interrupt void errISR_TPM2O() { for (;;)    asm BGND; }
interrupt void errISR_TPM2CH4() { for (;;)  asm BGND; }
```

```
interrupt void errISR_TPM2CH3() { for (;;)  asm BGND; }
interrupt void errISR_TPM2CH2() { for (;;)  asm BGND; }
interrupt void errISR_TPM2CH1() { for (;;)  asm BGND; }
interrupt void errISR_TPM2CH0() { for (;;)  asm BGND; }
interrupt void errISR_TPM1O() { for (;;)    asm BGND; }
interrupt void errISR_TPM1CH2() { for (;;)  asm BGND; }
interrupt void errISR_TPM1CH1() { for (;;)  asm BGND; }
interrupt void errISR_TPM1CH0() { for (;;)  asm BGND; }
interrupt void errISR_ICG() { for (;;)      asm BGND; }
interrupt void errISR_LowPower() { for (;;) asm BGND; }
interrupt void errISR_IRQ() { for (;;)      asm BGND; }
interrupt void errISR_SWI() { for (;;)      asm BGND; }
```

6.3.8 Project.prm

```
NAMES END /* Muss hier definiert sein.
             Kann fuer externe Objektdateien verwendet werden.*/

SEGMENTS
    Z_RAM                   =  READ_WRITE   0x0080 TO 0x00FF;  //   128 Bytes RAM
    RAM                     =  READ_WRITE   0x0100 TO 0x107F;  //  3968 Bytes RAM
    ROM                     =  READ_ONLY    0x182C TO 0xFFAF;  // 59268 Bytes ROM
    ROM1                    =  READ_ONLY    0x1080 TO 0x17FF;  //  1920 Bytes ROM
    ROM2                    =  READ_ONLY    0xFFC0 TO 0xFFCB;  //    12 Bytes ROM
(Unused Vector Space)
 /* INTVECTS                =  READ_ONLY    0xFFCC TO 0xFFFF; Reserved for Interrupt
Vectors */
END

PLACEMENT
  DEFAULT_ROM                INTO ROM;
  DEFAULT_RAM                INTO RAM;
  _DATA_ZEROPAGE, MY_ZEROPAGE INTO Z_RAM;
END

STACKSIZE 0x200      // Stacksize 0x200 => 512 Bytes

VECTOR ADDRESS 0xFFCC errISR_RTI        // RTI
VECTOR ADDRESS 0xFFCE errISR_IIC        // IIC
VECTOR ADDRESS 0xFFD0 errISR_ATD        // ATD conversion
VECTOR ADDRESS 0xFFD2 errISR_KBD        // Keyboard
VECTOR ADDRESS 0xFFD4 errISR_SCI2T      // SCI2 transmit
VECTOR ADDRESS 0xFFD6 errISR_SCI2R      // SCI2 receive
VECTOR ADDRESS 0xFFD8 errISR_SCI2E      // SCI2 error
VECTOR ADDRESS 0xFFDA errISR_SCI1T      // SCI1 transmit
VECTOR ADDRESS 0xFFDC errISR_SCI1R      // SCI1 receive
VECTOR ADDRESS 0xFFDE errISR_SCI1E      // SCI1 error
VECTOR ADDRESS 0xFFE0 errISR_SPI        // SPI
VECTOR ADDRESS 0xFFE2 errISR_TPM2O      // TPM2 overflow
VECTOR ADDRESS 0xFFE4 errISR_TPM2CH4    // TPM2 channel 4
VECTOR ADDRESS 0xFFE6 errISR_TPM2CH3    // TPM2 channel 3
VECTOR ADDRESS 0xFFE8 errISR_TPM2CH2    // TPM2 channel 2
VECTOR ADDRESS 0xFFEA errISR_TPM2CH1    // TPM2 channel 1
VECTOR ADDRESS 0xFFEC ISRToggleEdge    // TPM2 channel 0
VECTOR ADDRESS 0xFFEE errISR_TPM1O      // TPM1 overflow
VECTOR ADDRESS 0xFFF0 errISR_TPM1CH2    // TPM1 channel 2
VECTOR ADDRESS 0xFFF2 errISR_TPM1CH1    // TPM1 channel 1
VECTOR ADDRESS 0xFFF4 errISR_TPM1CH0    // TPM1 channel 0
VECTOR ADDRESS 0xFFF6 errISR_ICG        // ICG
VECTOR ADDRESS 0xFFF8 errISR_LowPower   // Low voltage detect
VECTOR ADDRESS 0xFFFA errISR_IRQ        // IRQ
VECTOR ADDRESS 0xFFFC errISR_SWI        // SWI
VECTOR ADDRESS 0xFFFE _Startup          // RESET
```

6.3.9 Project.map

```
PROGRAM "H:\HSLU\TA.MC\Unterlagen\Teil MC\Testatprojekt1\generator\bin\Project.abs"

********************************************************************************************
TARGET SECTION
--------------------------------------------------------------------------------------------
Processor   : Freescale HC08
Memory Model: SMALL
File Format : ELF\DWARF 2.0
Linker      : SmartLinker V-5.0.34 Build 8120, Apr 30 2008

********************************************************************************************
FILE SECTION
--------------------------------------------------------------------------------------------
main.c.o                                Model: SMALL,         Lang: ANSI-C
RTSHC08.C.o (ansiis.lib)                Model: SMALL,         Lang: ANSI-C
MC9S08GB60A.C.o                         Model: SMALL,         Lang: ANSI-C
Start08.c.o                             Model: SMALL,         Lang: ANSI-C
generator.c.o                           Model: SMALL,         Lang: ANSI-C
meter.c.o                               Model: SMALL,         Lang: ANSI-C

********************************************************************************************
STARTUP SECTION
--------------------------------------------------------------------------------------------
Entry point: 0x18A7 (_Startup)
_startupData is allocated at 0x1928 and uses 6 Bytes
extern struct _tagStartup {
  unsigne nofZeroOut     1
  _Range   pZeroOut       0x100     28
  _Copy    *toCopyDownBeg 0x1E89
} _startupData;

********************************************************************************************
SECTION-ALLOCATION SECTION
Section Name                     Size  Type     From       To       Segment
--------------------------------------------------------------------------------------------
.text                            1363     R     0x1936     0x1E88   ROM
.data                               4   R/W      0x100      0x103   RAM
.abs_section_0                      1   N/I        0x0        0x0   .absSeg0
.abs_section_1                      1   N/I        0x1        0x1   .absSeg1
.abs_section_2                      1   N/I        0x2        0x2   .absSeg2
.abs_section_3                      1   N/I        0x3        0x3   .absSeg3
.abs_section_4                      1   N/I        0x4        0x4   .absSeg4
.abs_section_5                      1   N/I        0x5        0x5   .absSeg5
.abs_section_6                      1   N/I        0x6        0x6   .absSeg6
.abs_section_7                      1   N/I        0x7        0x7   .absSeg7
.abs_section_8                      1   N/I        0x8        0x8   .absSeg8
.abs_section_9                      1   N/I        0x9        0x9   .absSeg9
.abs_section_a                      1   N/I        0xA        0xA   .absSeg10
.abs_section_b                      1   N/I        0xB        0xB   .absSeg11
.abs_section_c                      1   N/I        0xC        0xC   .absSeg12
.abs_section_d                      1   N/I        0xD        0xD   .absSeg13
.abs_section_e                      1   N/I        0xE        0xE   .absSeg14
.abs_section_f                      1   N/I        0xF        0xF   .absSeg15
.abs_section_10                     1   N/I       0x10       0x10   .absSeg16
.abs_section_11                     1   N/I       0x11       0x11   .absSeg17
.abs_section_12                     1   N/I       0x12       0x12   .absSeg18
.abs_section_13                     1   N/I       0x13       0x13   .absSeg19
.abs_section_14                     1   N/I       0x14       0x14   .absSeg20
.abs_section_16                     1   N/I       0x16       0x16   .absSeg21
.abs_section_17                     1   N/I       0x17       0x17   .absSeg22
.abs_section_1a                     1   N/I       0x1A       0x1A   .absSeg23
.abs_section_1b                     1   N/I       0x1B       0x1B   .absSeg24
.abs_section_1c                     1   N/I       0x1C       0x1C   .absSeg25
.abs_section_1d                     1   N/I       0x1D       0x1D   .absSeg26
.abs_section_1e                     1   N/I       0x1E       0x1E   .absSeg27
.abs_section_1f                     1   N/I       0x1F       0x1F   .absSeg28
.abs_section_22                     1   N/I       0x22       0x22   .absSeg29
.abs_section_23                     1   N/I       0x23       0x23   .absSeg30
.abs_section_24                     1   N/I       0x24       0x24   .absSeg31
```

```
.abs_section_25                 1   N/I       0x25       0x25   .absSeg32
.abs_section_26                 1   N/I       0x26       0x26   .absSeg33
.abs_section_27                 1   N/I       0x27       0x27   .absSeg34
.abs_section_28                 1   N/I       0x28       0x28   .absSeg35
.abs_section_29                 1   N/I       0x29       0x29   .absSeg36
.abs_section_2a                 1   N/I       0x2A       0x2A   .absSeg37
.abs_section_2b                 1   N/I       0x2B       0x2B   .absSeg38
.abs_section_2d                 1   N/I       0x2D       0x2D   .absSeg39
.abs_section_30                 1   N/I       0x30       0x30   .absSeg40
.abs_section_35                 1   N/I       0x35       0x35   .absSeg41
.abs_section_38                 1   N/I       0x38       0x38   .absSeg42
.abs_section_3b                 1   N/I       0x3B       0x3B   .absSeg43
.abs_section_40                 1   N/I       0x40       0x40   .absSeg44
.abs_section_41                 1   N/I       0x41       0x41   .absSeg45
.abs_section_42                 1   N/I       0x42       0x42   .absSeg46
.abs_section_43                 1   N/I       0x43       0x43   .absSeg47
.abs_section_44                 1   N/I       0x44       0x44   .absSeg48
.abs_section_45                 1   N/I       0x45       0x45   .absSeg49
.abs_section_46                 1   N/I       0x46       0x46   .absSeg50
.abs_section_47                 1   N/I       0x47       0x47   .absSeg51
.abs_section_48                 1   N/I       0x48       0x48   .absSeg52
.abs_section_49                 1   N/I       0x49       0x49   .absSeg53
.abs_section_4a                 1   N/I       0x4A       0x4A   .absSeg54
.abs_section_4b                 1   N/I       0x4B       0x4B   .absSeg55
.abs_section_4e                 1   N/I       0x4E       0x4E   .absSeg56
.abs_section_50                 1   N/I       0x50       0x50   .absSeg57
.abs_section_51                 1   N/I       0x51       0x51   .absSeg58
.abs_section_54                 1   N/I       0x54       0x54   .absSeg59
.abs_section_58                 1   N/I       0x58       0x58   .absSeg60
.abs_section_59                 1   N/I       0x59       0x59   .absSeg61
.abs_section_5a                 1   N/I       0x5A       0x5A   .absSeg62
.abs_section_5b                 1   N/I       0x5B       0x5B   .absSeg63
.abs_section_5c                 1   N/I       0x5C       0x5C   .absSeg64
.abs_section_60                 1   N/I       0x60       0x60   .absSeg65
.abs_section_65                 1   N/I       0x65       0x65   .absSeg66
.abs_section_68                 1   N/I       0x68       0x68   .absSeg67
.abs_section_6b                 1   N/I       0x6B       0x6B   .absSeg68
.abs_section_6e                 1   N/I       0x6E       0x6E   .absSeg69
.abs_section_71                 1   N/I       0x71       0x71   .absSeg70
.abs_section_1800               1   N/I     0x1800     0x1800   .absSeg71
.abs_section_1801               1   N/I     0x1801     0x1801   .absSeg72
.abs_section_1802               1   N/I     0x1802     0x1802   .absSeg73
.abs_section_1808               1   N/I     0x1808     0x1808   .absSeg74
.abs_section_1809               1   N/I     0x1809     0x1809   .absSeg75
.abs_section_180a               1   N/I     0x180A     0x180A   .absSeg76
.abs_section_1810               1   N/I     0x1810     0x1810   .absSeg77
.abs_section_1811               1   N/I     0x1811     0x1811   .absSeg78
.abs_section_1812               1   N/I     0x1812     0x1812   .absSeg79
.abs_section_1813               1   N/I     0x1813     0x1813   .absSeg80
.abs_section_1816               1   N/I     0x1816     0x1816   .absSeg81
.abs_section_1817               1   N/I     0x1817     0x1817   .absSeg82
.abs_section_1818               1   N/I     0x1818     0x1818   .absSeg83
.abs_section_1820               1   N/I     0x1820     0x1820   .absSeg84
.abs_section_1821               1   N/I     0x1821     0x1821   .absSeg85
.abs_section_1823               1   N/I     0x1823     0x1823   .absSeg86
.abs_section_1824               1   N/I     0x1824     0x1824   .absSeg87
.abs_section_1825               1   N/I     0x1825     0x1825   .absSeg88
.abs_section_1826               1   N/I     0x1826     0x1826   .absSeg89
.abs_section_18                 2   N/I       0x18       0x19   .absSeg90
.abs_section_20                 2   N/I       0x20       0x21   .absSeg91
.abs_section_31                 2   N/I       0x31       0x32   .absSeg92
.abs_section_33                 2   N/I       0x33       0x34   .absSeg93
.abs_section_36                 2   N/I       0x36       0x37   .absSeg94
.abs_section_39                 2   N/I       0x39       0x3A   .absSeg95
.abs_section_3c                 2   N/I       0x3C       0x3D   .absSeg96
.abs_section_4c                 2   N/I       0x4C       0x4D   .absSeg97
.abs_section_52                 2   N/I       0x52       0x53   .absSeg98
.abs_section_61                 2   N/I       0x61       0x62   .absSeg99
.abs_section_63                 2   N/I       0x63       0x64   .absSeg100
.abs_section_66                 2   N/I       0x66       0x67   .absSeg101
.abs_section_69                 2   N/I       0x69       0x6A   .absSeg102
.abs_section_6c                 2   N/I       0x6C       0x6D   .absSeg103
```

```
.abs_section_6f                    2   N/I       0x6F       0x70   .absSeg104
.abs_section_72                    2   N/I       0x72       0x73   .absSeg105
.abs_section_1806                  2   N/I     0x1806     0x1807   .absSeg106
.abs_section_1814                  2   N/I     0x1814     0x1815   .absSeg107
.bss                              24   R/W      0x104      0x11B   RAM
.startData                        14     R     0x1928     0x1935   ROM
.init                            252     R     0x182C     0x1927   ROM
.stack                           512   R/W      0x11C      0x31B   RAM
.copy                              8     R     0x1E89     0x1E90   ROM
.vectSeg108_vect                  52     R     0xFFCC     0xFFFF   .vectSeg108

Summary of section sizes per section type:
READ_ONLY (R):         699 (dec:     1689)
READ_WRITE (R/W):      21C (dec:      540)
NO_INIT (N/I):          7E (dec:      126)

*****************************************************************************************
VECTOR-ALLOCATION SECTION
    Address     InitValue   InitFunction
-----------------------------------------------------------------------------------------
     0xFFCC        0x18B0   errISR_RTI
     0xFFCE        0x18B5   errISR_IIC
     0xFFD0        0x18BA   errISR_ATD
     0xFFD2        0x18BF   errISR_KBD
     0xFFD4        0x18C4   errISR_SCI2T
     0xFFD6        0x18C9   errISR_SCI2R
     0xFFD8        0x18CE   errISR_SCI2E
     0xFFDA        0x18D3   errISR_SCI1T
     0xFFDC        0x18D8   errISR_SCI1R
     0xFFDE        0x18DD   errISR_SCI1E
     0xFFE0        0x18E2   errISR_SPI
     0xFFE2        0x18E7   errISR_TPM2O
     0xFFE4        0x18EC   errISR_TPM2CH4
     0xFFE6        0x18F1   errISR_TPM2CH3
     0xFFE8        0x18F6   errISR_TPM2CH2
     0xFFEA        0x18FB   errISR_TPM2CH1
     0xFFEC        0x1D2E   ISRToggleEdge
     0xFFEE        0x1900   errISR_TPM1O
     0xFFF0        0x1905   errISR_TPM1CH2
     0xFFF2        0x190A   errISR_TPM1CH1
     0xFFF4        0x190F   errISR_TPM1CH0
     0xFFF6        0x1914   errISR_ICG
     0xFFF8        0x1919   errISR_LowPower
     0xFFFA        0x191E   errISR_IRQ
     0xFFFC        0x1923   errISR_SWI
     0xFFFE        0x18A7   _Startup

*****************************************************************************************
OBJECT-ALLOCATION SECTION
     Name               Module                 Addr   hSize   dSize     Ref    Section   RLIB
-----------------------------------------------------------------------------------------
MODULE:                 -- main.c.o --
- PROCEDURES:
     initPorts                                 1936      10      16       1   .text
     main                                      1946      19      25       1   .text
- VARIABLES:
MODULE:                 -- RTSHC08.C.o (ansiis.lib) --
- PROCEDURES:
     _PUSH_ARGS_L                              195F      20      32       3   .text
     _ENTER_UNARY_L                            197F      29      41       2   .text
     _ENTER_BINARY_L                           19A8      16      22       5   .text
     _ENTER_BINARY_L_RC                        19BE      19      25       4   .text
     _ENTER_BINARY_L_LC                        19D7      1A      26       1   .text
     _LADD_k_is_k_plus_j                       19F1      1F      31       1   .text
     _k_is_k_plus_j_l                          1A10      1E      30       2   .text
     _k_is_k_plus_j_i                          1A2E      12      18       3   .text
     _LSUB_k_is_k_minus_j                      1A40      1F      31       1   .text
     _LMUL_k_is_k_mul_j                        1A5F      81     129       2   .text
     _LDIVMOD                                  1AE0      A4     164       3   .text
     _NEG_L_HX                                 1B84      13      19       2   .text
     _ABS_L_HX                                 1B97       6       6       2   .text
     _SPLITSIGN_L                              1B9D       A      10       1   .text
```

```
     _LMODU_k_is_k_mod_j                    1BA7       B      11       1   .text
     _LDIVU_k_is_k_div_j                    1BB2      17      23       2   .text
     _LDIVS_k_is_k_div_j                    1BC9      27      39       2   .text
     _LCMP_k_rel_j                          1BF0      1F      31       1   .text
     _LDEC                                  1C0F      1C      28       1   .text
     _LADD                                  1C2B       6       6       3   .text
     _LSUB                                  1C31       6       6       4   .text
     _LMUL                                  1C37       6       6       2   .text
     _LMUL_RC                               1C3D       6       6       1   .text
     _LDIVS_LC                              1C43       6       6       2   .text
     _LDIVS_RC                              1C49       6       6       1   .text
     _LDIVU                                 1C4F       6       6       3   .text
     _LDIVU_RC                              1C55       6       6       3   .text
     _LMODU_RC                              1C5B       6       6       3   .text
     _LLSL                                  1C61      16      22       3   .text
     _LCMP                                  1C77       6       6       2   .text
     _POP32                                 1C7D      1A      26       5   .text
- VARIABLES:
MODULE:                 -- MC9S08GB60A.C.o --
- PROCEDURES:
- VARIABLES:
     _PTAD                                     0       1       1       2   .abs_section_0
     _PTAPE                                    1       1       1       0   .abs_section_1
     _PTASE                                    2       1       1       0   .abs_section_2
     _PTADD                                    3       1       1       1   .abs_section_3
     _PTBD                                     4       1       1       2   .abs_section_4
     _PTBPE                                    5       1       1       0   .abs_section_5
     _PTBSE                                    6       1       1       0   .abs_section_6
     _PTBDD                                    7       1       1       1   .abs_section_7
     _PTCD                                     8       1       1       0   .abs_section_8
     _PTCPE                                    9       1       1       0   .abs_section_9
     _PTCSE                                    A       1       1       0   .abs_section_a
     _PTCDD                                    B       1       1       0   .abs_section_b
     _PTDD                                     C       1       1       1   .abs_section_c
     _PTDPE                                    D       1       1       0   .abs_section_d
     _PTDSE                                    E       1       1       0   .abs_section_e
     _PTDDD                                    F       1       1       0   .abs_section_f
     _PTED                                    10       1       1       0   .abs_section_10
     _PTEPE                                   11       1       1       0   .abs_section_11
     _PTESE                                   12       1       1       0   .abs_section_12
     _PTEDD                                   13       1       1       0   .abs_section_13
     _IRQSC                                   14       1       1       0   .abs_section_14
     _KBI1SC                                  16       1       1       0   .abs_section_16
     _KBI1PE                                  17       1       1       0   .abs_section_17
     _SCI1C1                                  1A       1       1       0   .abs_section_1a
     _SCI1C2                                  1B       1       1       0   .abs_section_1b
     _SCI1S1                                  1C       1       1       0   .abs_section_1c
     _SCI1S2                                  1D       1       1       0   .abs_section_1d
     _SCI1C3                                  1E       1       1       0   .abs_section_1e
     _SCI1D                                   1F       1       1       0   .abs_section_1f
     _SCI2C1                                  22       1       1       0   .abs_section_22
     _SCI2C2                                  23       1       1       0   .abs_section_23
     _SCI2S1                                  24       1       1       0   .abs_section_24
     _SCI2S2                                  25       1       1       0   .abs_section_25
     _SCI2C3                                  26       1       1       0   .abs_section_26
     _SCI2D                                   27       1       1       0   .abs_section_27
     _SPI1C1                                  28       1       1       0   .abs_section_28
     _SPI1C2                                  29       1       1       0   .abs_section_29
     _SPI1BR                                  2A       1       1       0   .abs_section_2a
     _SPI1S                                   2B       1       1       0   .abs_section_2b
     _SPI1D                                   2D       1       1       0   .abs_section_2d
     _TPM1SC                                  30       1       1       2   .abs_section_30
     _TPM1C0SC                                35       1       1       2   .abs_section_35
     _TPM1C1SC                                38       1       1       0   .abs_section_38
     _TPM1C2SC                                3B       1       1       0   .abs_section_3b
     _PTFD                                    40       1       1       2   .abs_section_40
     _PTFPE                                   41       1       1       0   .abs_section_41
     _PTFSE                                   42       1       1       0   .abs_section_42
     _PTFDD                                   43       1       1       1   .abs_section_43
     _PTGD                                    44       1       1       0   .abs_section_44
     _PTGPE                                   45       1       1       0   .abs_section_45
     _PTGSE                                   46       1       1       0   .abs_section_46
```

```
     _PTGDD                                47       1       1       0   .abs_section_47
     _ICGC1                                48       1       1       1   .abs_section_48
     _ICGC2                                49       1       1       1   .abs_section_49
     _ICGS1                                4A       1       1       1   .abs_section_4a
     _ICGS2                                4B       1       1       0   .abs_section_4b
     _ICGTRM                               4E       1       1       0   .abs_section_4e
     _ATD1C                                50       1       1       0   .abs_section_50
     _ATD1SC                               51       1       1       0   .abs_section_51
     _ATD1PE                               54       1       1       0   .abs_section_54
     _IIC1A                                58       1       1       0   .abs_section_58
     _IIC1F                                59       1       1       0   .abs_section_59
     _IIC1C                                5A       1       1       0   .abs_section_5a
     _IIC1S                                5B       1       1       0   .abs_section_5b
     _IIC1D                                5C       1       1       0   .abs_section_5c
     _TPM2SC                               60       1       1       5   .abs_section_60
     _TPM2C0SC                             65       1       1       6   .abs_section_65
     _TPM2C1SC                             68       1       1       0   .abs_section_68
     _TPM2C2SC                             6B       1       1       0   .abs_section_6b
     _TPM2C3SC                             6E       1       1       0   .abs_section_6e
     _TPM2C4SC                             71       1       1       0   .abs_section_71
     _SRS                                1800       1       1       0
.abs_section_1800
     _SBDFR                              1801       1       1       0
.abs_section_1801
     _SOPT                               1802       1       1       1
.abs_section_1802
     _SRTISC                             1808       1       1       0
.abs_section_1808
     _SPMSC1                             1809       1       1       0
.abs_section_1809
     _SPMSC2                             180A       1       1       0
.abs_section_180a
     _DBGCAH                             1810       1       1       0
.abs_section_1810
     _DBGCAL                             1811       1       1       0
.abs_section_1811
     _DBGCBH                             1812       1       1       0
.abs_section_1812
     _DBGCBL                             1813       1       1       0
.abs_section_1813
     _DBGC                               1816       1       1       0
.abs_section_1816
     _DBGT                               1817       1       1       0
.abs_section_1817
     _DBGS                               1818       1       1       0
.abs_section_1818
     _FCDIV                              1820       1       1       0
.abs_section_1820
     _FOPT                               1821       1       1       0
.abs_section_1821
     _FCNFG                              1823       1       1       0
.abs_section_1823
     _FPROT                              1824       1       1       0
.abs_section_1824
     _FSTAT                              1825       1       1       0
.abs_section_1825
     _FCMD                               1826       1       1       0
.abs_section_1826
     _SCI1BD                               18       2       2       0   .abs_section_18
     _SCI2BD                               20       2       2       0   .abs_section_20
     _TPM1CNT                              31       2       2       0   .abs_section_31
     _TPM1MOD                              33       2       2       1   .abs_section_33
     _TPM1C0V                              36       2       2       1   .abs_section_36
     _TPM1C1V                              39       2       2       0   .abs_section_39
     _TPM1C2V                              3C       2       2       0   .abs_section_3c
     _ICGFLT                               4C       2       2       0   .abs_section_4c
     _ATD1R                                52       2       2       0   .abs_section_52
     _TPM2CNT                              61       2       2       1   .abs_section_61
     _TPM2MOD                              63       2       2       1   .abs_section_63
     _TPM2C0V                              66       2       2       2   .abs_section_66
     _TPM2C1V                              69       2       2       0   .abs_section_69
     _TPM2C2V                              6C       2       2       0   .abs_section_6c
```

```
     _TPM2C3V                                  6F       2       2       0   .abs_section_6f
     _TPM2C4V                                  72       2       2       0   .abs_section_72
     _SDID                                   1806       2       2       0
.abs_section_1806
     _DBGF                                   1814       2       2       0
.abs_section_1814
MODULE:                 -- Start08.c.o --
- PROCEDURES:
     loadByte                                182C       E      14       5   .init
     Init                                    183A      6D     109       1   .init
     _Startup                                18A7       9       9       0   .init
     errISR_RTI                              18B0       5       5       0   .init
     errISR_IIC                              18B5       5       5       0   .init
     errISR_ATD                              18BA       5       5       0   .init
     errISR_KBD                              18BF       5       5       0   .init
     errISR_SCI2T                            18C4       5       5       0   .init
     errISR_SCI2R                            18C9       5       5       0   .init
     errISR_SCI2E                            18CE       5       5       0   .init
     errISR_SCI1T                            18D3       5       5       0   .init
     errISR_SCI1R                            18D8       5       5       0   .init
     errISR_SCI1E                            18DD       5       5       0   .init
     errISR_SPI                              18E2       5       5       0   .init
     errISR_TPM2O                            18E7       5       5       0   .init
     errISR_TPM2CH4                          18EC       5       5       0   .init
     errISR_TPM2CH3                          18F1       5       5       0   .init
     errISR_TPM2CH2                          18F6       5       5       0   .init
     errISR_TPM2CH1                          18FB       5       5       0   .init
     errISR_TPM1O                            1900       5       5       0   .init
     errISR_TPM1CH2                          1905       5       5       0   .init
     errISR_TPM1CH1                          190A       5       5       0   .init
     errISR_TPM1CH0                          190F       5       5       0   .init
     errISR_ICG                              1914       5       5       0   .init
     errISR_LowPower                         1919       5       5       0   .init
     errISR_IRQ                              191E       5       5       0   .init
     errISR_SWI                              1923       5       5       0   .init
- VARIABLES:
     _startupData                            1928       6       6       4   .startData
- LABELS:
     __SEG_END_SSTACK                         31C       0       0       1
MODULE:                 -- generator.c.o --
- PROCEDURES:
     initClock                               1C97       F      15       1   .text
     initGenerator                           1CA6      13      19       1   .text
     setGenerator                            1CB9      59      89       1   .text
- VARIABLES:
MODULE:                 -- meter.c.o --
- PROCEDURES:
     initTimer                               1D12      11      17       1   .text
     initInputCapture                        1D23       B      11       1   .text
     ISRToggleEdge                           1D2E      78     120       0   .text
     show                                    1DA6      E3     227       1   .text
- VARIABLES:
     CyclesPerSecond                          100       4       4       3   .data
     TimeHighCycle                            104       4       4       8   .bss
     TimeLowCycle                             108       4       4       8   .bss
     HighDifference                           10C       4       4       5   .bss
     LowDifference                            110       4       4       5   .bss
     TimeMicroSecond                          114       4       4       9   .bss
     DutyCycle                                118       4       4       2   .bss

*****************************************************************************************
MODULE STATISTIC
  Name                                    Data   Code  Const
-----------------------------------------------------------------------------------------
  main.c.o                                   0     41      0
  RTSHC08.C.o (ansiis.lib)                   0    824      0
  MC9S08GB60A.C.o                          126      0      0
  Start08.c.o                                0    252      0
  generator.c.o                              0    123      0
  meter.c.o                                 28    375      0
  other                                    512     66      8
```

```
*****************************************************************************************
SECTION USE IN OBJECT-ALLOCATION SECTION
-----------------------------------------------------------------------------------------
SECTION: ".text"
  initPorts main _PUSH_ARGS_L _ENTER_UNARY_L _ENTER_BINARY_L
  _ENTER_BINARY_L_RC _ENTER_BINARY_L_LC _LADD_k_is_k_plus_j _k_is_k_plus_j_l
  _k_is_k_plus_j_i _LSUB_k_is_k_minus_j _LMUL_k_is_k_mul_j _LDIVMOD _NEG_L_HX
  _ABS_L_HX _SPLITSIGN_L _LMODU_k_is_k_mod_j _LDIVU_k_is_k_div_j
  _LDIVS_k_is_k_div_j _LCMP_k_rel_j _LDEC _LADD _LSUB _LMUL _LMUL_RC _LDIVS_LC
  _LDIVS_RC _LDIVU _LDIVU_RC _LMODU_RC _LLSL _LCMP _POP32 initClock
  initGenerator setGenerator initTimer initInputCapture ISRToggleEdge show
SECTION: ".data"
  CyclesPerSecond
SECTION: ".bss"
  TimeHighCycle TimeLowCycle HighDifference LowDifference TimeMicroSecond
  DutyCycle
SECTION: ".init"
  loadByte Init _Startup errISR_RTI errISR_IIC errISR_ATD errISR_KBD
  errISR_SCI2T errISR_SCI2R errISR_SCI2E errISR_SCI1T errISR_SCI1R errISR_SCI1E
  errISR_SPI errISR_TPM2O errISR_TPM2CH4 errISR_TPM2CH3 errISR_TPM2CH2
  errISR_TPM2CH1 errISR_TPM1O errISR_TPM1CH2 errISR_TPM1CH1 errISR_TPM1CH0
  errISR_ICG errISR_LowPower errISR_IRQ errISR_SWI
SECTION: ".abs_section_0"
  _PTAD
SECTION: ".abs_section_1"
  _PTAPE
SECTION: ".abs_section_2"
  _PTASE
SECTION: ".abs_section_3"
  _PTADD
SECTION: ".abs_section_4"
  _PTBD
SECTION: ".abs_section_5"
  _PTBPE
SECTION: ".abs_section_6"
  _PTBSE
SECTION: ".abs_section_7"
  _PTBDD
SECTION: ".abs_section_8"
  _PTCD
SECTION: ".abs_section_9"
  _PTCPE
SECTION: ".abs_section_a"
  _PTCSE
SECTION: ".abs_section_b"
  _PTCDD
SECTION: ".abs_section_c"
  _PTDD
SECTION: ".abs_section_d"
  _PTDPE
SECTION: ".abs_section_e"
  _PTDSE
SECTION: ".abs_section_f"
  _PTDDD
SECTION: ".abs_section_10"
  _PTED
SECTION: ".abs_section_11"
  _PTEPE
SECTION: ".abs_section_12"
  _PTESE
SECTION: ".abs_section_13"
  _PTEDD
SECTION: ".abs_section_14"
  _IRQSC
SECTION: ".abs_section_16"
  _KBI1SC
SECTION: ".abs_section_17"
  _KBI1PE
SECTION: ".abs_section_1a"
  _SCI1C1
SECTION: ".abs_section_1b"
  _SCI1C2
```

```
SECTION: ".abs_section_1c"
  _SCI1S1
SECTION: ".abs_section_1d"
  _SCI1S2
SECTION: ".abs_section_1e"
  _SCI1C3
SECTION: ".abs_section_1f"
  _SCI1D
SECTION: ".abs_section_22"
  _SCI2C1
SECTION: ".abs_section_23"
  _SCI2C2
SECTION: ".abs_section_24"
  _SCI2S1
SECTION: ".abs_section_25"
  _SCI2S2
SECTION: ".abs_section_26"
  _SCI2C3
SECTION: ".abs_section_27"
  _SCI2D
SECTION: ".abs_section_28"
  _SPI1C1
SECTION: ".abs_section_29"
  _SPI1C2
SECTION: ".abs_section_2a"
  _SPI1BR
SECTION: ".abs_section_2b"
  _SPI1S
SECTION: ".abs_section_2d"
  _SPI1D
SECTION: ".abs_section_30"
  _TPM1SC
SECTION: ".abs_section_35"
  _TPM1C0SC
SECTION: ".abs_section_38"
  _TPM1C1SC
SECTION: ".abs_section_3b"
  _TPM1C2SC
SECTION: ".abs_section_40"
  _PTFD
SECTION: ".abs_section_41"
  _PTFPE
SECTION: ".abs_section_42"
  _PTFSE
SECTION: ".abs_section_43"
  _PTFDD
SECTION: ".abs_section_44"
  _PTGD
SECTION: ".abs_section_45"
  _PTGPE
SECTION: ".abs_section_46"
  _PTGSE
SECTION: ".abs_section_47"
  _PTGDD
SECTION: ".abs_section_48"
  _ICGC1
SECTION: ".abs_section_49"
  _ICGC2
SECTION: ".abs_section_4a"
  _ICGS1
SECTION: ".abs_section_4b"
  _ICGS2
SECTION: ".abs_section_4e"
  _ICGTRM
SECTION: ".abs_section_50"
  _ATD1C
SECTION: ".abs_section_51"
  _ATD1SC
SECTION: ".abs_section_54"
  _ATD1PE
SECTION: ".abs_section_58"
  _IIC1A
```

```
SECTION: ".abs_section_59"
  _IIC1F
SECTION: ".abs_section_5a"
  _IIC1C
SECTION: ".abs_section_5b"
  _IIC1S
SECTION: ".abs_section_5c"
  _IIC1D
SECTION: ".abs_section_60"
  _TPM2SC
SECTION: ".abs_section_65"
  _TPM2C0SC
SECTION: ".abs_section_68"
  _TPM2C1SC
SECTION: ".abs_section_6b"
  _TPM2C2SC
SECTION: ".abs_section_6e"
  _TPM2C3SC
SECTION: ".abs_section_71"
  _TPM2C4SC
SECTION: ".abs_section_1800"
  _SRS
SECTION: ".abs_section_1801"
  _SBDFR
SECTION: ".abs_section_1802"
  _SOPT
SECTION: ".abs_section_1808"
  _SRTISC
SECTION: ".abs_section_1809"
  _SPMSC1
SECTION: ".abs_section_180a"
  _SPMSC2
SECTION: ".abs_section_1810"
  _DBGCAH
SECTION: ".abs_section_1811"
  _DBGCAL
SECTION: ".abs_section_1812"
  _DBGCBH
SECTION: ".abs_section_1813"
  _DBGCBL
SECTION: ".abs_section_1816"
  _DBGC
SECTION: ".abs_section_1817"
  _DBGT
SECTION: ".abs_section_1818"
  _DBGS
SECTION: ".abs_section_1820"
  _FCDIV
SECTION: ".abs_section_1821"
  _FOPT
SECTION: ".abs_section_1823"
  _FCNFG
SECTION: ".abs_section_1824"
  _FPROT
SECTION: ".abs_section_1825"
  _FSTAT
SECTION: ".abs_section_1826"
  _FCMD
SECTION: ".abs_section_18"
  _SCI1BD
SECTION: ".abs_section_20"
  _SCI2BD
SECTION: ".abs_section_31"
  _TPM1CNT
SECTION: ".abs_section_33"
  _TPM1MOD
SECTION: ".abs_section_36"
  _TPM1C0V
SECTION: ".abs_section_39"
  _TPM1C1V
SECTION: ".abs_section_3c"
  _TPM1C2V
```

```
SECTION: ".abs_section_4c"
  _ICGFLT
SECTION: ".abs_section_52"
  _ATD1R
SECTION: ".abs_section_61"
  _TPM2CNT
SECTION: ".abs_section_63"
  _TPM2MOD
SECTION: ".abs_section_66"
  _TPM2C0V
SECTION: ".abs_section_69"
  _TPM2C1V
SECTION: ".abs_section_6c"
  _TPM2C2V
SECTION: ".abs_section_6f"
  _TPM2C3V
SECTION: ".abs_section_72"
  _TPM2C4V
SECTION: ".abs_section_1806"
  _SDID
SECTION: ".abs_section_1814"
  _DBGF

*******************************************************************************************
OBJECT LIST SORTED BY ADDRESS
     Name                                        Addr    hSize   dSize     Ref    Section   RLIB
-------------------------------------------------------------------------------------------
     _PTAD                                          0       1       1       2   .abs_section_0
     _PTAPE                                         1       1       1       0   .abs_section_1
     _PTASE                                         2       1       1       0   .abs_section_2
     _PTADD                                         3       1       1       1   .abs_section_3
     _PTBD                                          4       1       1       2   .abs_section_4
     _PTBPE                                         5       1       1       0   .abs_section_5
     _PTBSE                                         6       1       1       0   .abs_section_6
     _PTBDD                                         7       1       1       1   .abs_section_7
     _PTCD                                          8       1       1       0   .abs_section_8
     _PTCPE                                         9       1       1       0   .abs_section_9
     _PTCSE                                         A       1       1       0   .abs_section_a
     _PTCDD                                         B       1       1       0   .abs_section_b
     _PTDD                                          C       1       1       1   .abs_section_c
     _PTDPE                                         D       1       1       0   .abs_section_d
     _PTDSE                                         E       1       1       0   .abs_section_e
     _PTDDD                                         F       1       1       0   .abs_section_f
     _PTED                                         10       1       1       0   .abs_section_10
     _PTEPE                                        11       1       1       0   .abs_section_11
     _PTESE                                        12       1       1       0   .abs_section_12
     _PTEDD                                        13       1       1       0   .abs_section_13
     _IRQSC                                        14       1       1       0   .abs_section_14
     _KBI1SC                                       16       1       1       0   .abs_section_16
     _KBI1PE                                       17       1       1       0   .abs_section_17
     _SCI1BD                                       18       2       2       0   .abs_section_18
     _SCI1C1                                       1A       1       1       0   .abs_section_1a
     _SCI1C2                                       1B       1       1       0   .abs_section_1b
     _SCI1S1                                       1C       1       1       0   .abs_section_1c
     _SCI1S2                                       1D       1       1       0   .abs_section_1d
     _SCI1C3                                       1E       1       1       0   .abs_section_1e
     _SCI1D                                        1F       1       1       0   .abs_section_1f
     _SCI2BD                                       20       2       2       0   .abs_section_20
     _SCI2C1                                       22       1       1       0   .abs_section_22
     _SCI2C2                                       23       1       1       0   .abs_section_23
     _SCI2S1                                       24       1       1       0   .abs_section_24
     _SCI2S2                                       25       1       1       0   .abs_section_25
     _SCI2C3                                       26       1       1       0   .abs_section_26
     _SCI2D                                        27       1       1       0   .abs_section_27
     _SPI1C1                                       28       1       1       0   .abs_section_28
     _SPI1C2                                       29       1       1       0   .abs_section_29
     _SPI1BR                                       2A       1       1       0   .abs_section_2a
     _SPI1S                                        2B       1       1       0   .abs_section_2b
     _SPI1D                                        2D       1       1       0   .abs_section_2d
     _TPM1SC                                       30       1       1       2   .abs_section_30
     _TPM1CNT                                      31       2       2       0   .abs_section_31
     _TPM1MOD                                      33       2       2       1   .abs_section_33
```

```
     _TPM1C0SC                      35       1       1       2   .abs_section_35
     _TPM1C0V                       36       2       2       1   .abs_section_36
     _TPM1C1SC                      38       1       1       0   .abs_section_38
     _TPM1C1V                       39       2       2       0   .abs_section_39
     _TPM1C2SC                      3B       1       1       0   .abs_section_3b
     _TPM1C2V                       3C       2       2       0   .abs_section_3c
     _PTFD                          40       1       1       2   .abs_section_40
     _PTFPE                         41       1       1       0   .abs_section_41
     _PTFSE                         42       1       1       0   .abs_section_42
     _PTFDD                         43       1       1       1   .abs_section_43
     _PTGD                          44       1       1       0   .abs_section_44
     _PTGPE                         45       1       1       0   .abs_section_45
     _PTGSE                         46       1       1       0   .abs_section_46
     _PTGDD                         47       1       1       0   .abs_section_47
     _ICGC1                         48       1       1       1   .abs_section_48
     _ICGC2                         49       1       1       1   .abs_section_49
     _ICGS1                         4A       1       1       1   .abs_section_4a
     _ICGS2                         4B       1       1       0   .abs_section_4b
     _ICGFLT                        4C       2       2       0   .abs_section_4c
     _ICGTRM                        4E       1       1       0   .abs_section_4e
     _ATD1C                         50       1       1       0   .abs_section_50
     _ATD1SC                        51       1       1       0   .abs_section_51
     _ATD1R                         52       2       2       0   .abs_section_52
     _ATD1PE                        54       1       1       0   .abs_section_54
     _IIC1A                         58       1       1       0   .abs_section_58
     _IIC1F                         59       1       1       0   .abs_section_59
     _IIC1C                         5A       1       1       0   .abs_section_5a
     _IIC1S                         5B       1       1       0   .abs_section_5b
     _IIC1D                         5C       1       1       0   .abs_section_5c
     _TPM2SC                        60       1       1       5   .abs_section_60
     _TPM2CNT                       61       2       2       1   .abs_section_61
     _TPM2MOD                       63       2       2       1   .abs_section_63
     _TPM2C0SC                      65       1       1       6   .abs_section_65
     _TPM2C0V                       66       2       2       2   .abs_section_66
     _TPM2C1SC                      68       1       1       0   .abs_section_68
     _TPM2C1V                       69       2       2       0   .abs_section_69
     _TPM2C2SC                      6B       1       1       0   .abs_section_6b
     _TPM2C2V                       6C       2       2       0   .abs_section_6c
     _TPM2C3SC                      6E       1       1       0   .abs_section_6e
     _TPM2C3V                       6F       2       2       0   .abs_section_6f
     _TPM2C4SC                      71       1       1       0   .abs_section_71
     _TPM2C4V                       72       2       2       0   .abs_section_72
     CyclesPerSecond               100       4       4       3   .data
     TimeHighCycle                 104       4       4       8   .bss
     TimeLowCycle                  108       4       4       8   .bss
     HighDifference                10C       4       4       5   .bss
     LowDifference                 110       4       4       5   .bss
     TimeMicroSecond               114       4       4       9   .bss
     DutyCycle                     118       4       4       2   .bss
     _SRS                         1800       1       1       0
.abs_section_1800
     _SBDFR                       1801       1       1       0
.abs_section_1801
     _SOPT                        1802       1       1       1
.abs_section_1802
     _SDID                        1806       2       2       0
.abs_section_1806
     _SRTISC                      1808       1       1       0
.abs_section_1808
     _SPMSC1                      1809       1       1       0
.abs_section_1809
     _SPMSC2                      180A       1       1       0
.abs_section_180a
     _DBGCAH                      1810       1       1       0
.abs_section_1810
     _DBGCAL                      1811       1       1       0
.abs_section_1811
     _DBGCBH                      1812       1       1       0
.abs_section_1812
     _DBGCBL                      1813       1       1       0
.abs_section_1813
     _DBGF                        1814       2       2       0
```

```
.abs_section_1814
     _DBGC                                 1816       1       1       0
.abs_section_1816
     _DBGT                                 1817       1       1       0
.abs_section_1817
     _DBGS                                 1818       1       1       0
.abs_section_1818
     _FCDIV                                1820       1       1       0
.abs_section_1820
     _FOPT                                 1821       1       1       0
.abs_section_1821
     _FCNFG                                1823       1       1       0
.abs_section_1823
     _FPROT                                1824       1       1       0
.abs_section_1824
     _FSTAT                                1825       1       1       0
.abs_section_1825
     _FCMD                                 1826       1       1       0
.abs_section_1826
     loadByte                              182C       E      14       5   .init
     Init                                  183A      6D     109       1   .init
     _Startup                              18A7       9       9       0   .init
     errISR_RTI                            18B0       5       5       0   .init
     errISR_IIC                            18B5       5       5       0   .init
     errISR_ATD                            18BA       5       5       0   .init
     errISR_KBD                            18BF       5       5       0   .init
     errISR_SCI2T                          18C4       5       5       0   .init
     errISR_SCI2R                          18C9       5       5       0   .init
     errISR_SCI2E                          18CE       5       5       0   .init
     errISR_SCI1T                          18D3       5       5       0   .init
     errISR_SCI1R                          18D8       5       5       0   .init
     errISR_SCI1E                          18DD       5       5       0   .init
     errISR_SPI                            18E2       5       5       0   .init
     errISR_TPM2O                          18E7       5       5       0   .init
     errISR_TPM2CH4                        18EC       5       5       0   .init
     errISR_TPM2CH3                        18F1       5       5       0   .init
     errISR_TPM2CH2                        18F6       5       5       0   .init
     errISR_TPM2CH1                        18FB       5       5       0   .init
     errISR_TPM1O                          1900       5       5       0   .init
     errISR_TPM1CH2                        1905       5       5       0   .init
     errISR_TPM1CH1                        190A       5       5       0   .init
     errISR_TPM1CH0                        190F       5       5       0   .init
     errISR_ICG                            1914       5       5       0   .init
     errISR_LowPower                       1919       5       5       0   .init
     errISR_IRQ                            191E       5       5       0   .init
     errISR_SWI                            1923       5       5       0   .init
     initPorts                             1936      10      16       1   .text
     main                                  1946      19      25       1   .text
     _PUSH_ARGS_L                          195F      20      32       3   .text
     _ENTER_UNARY_L                        197F      29      41       2   .text
     _ENTER_BINARY_L                       19A8      16      22       5   .text
     _ENTER_BINARY_L_RC                    19BE      19      25       4   .text
     _ENTER_BINARY_L_LC                    19D7      1A      26       1   .text
     _LADD_k_is_k_plus_j                   19F1      1F      31       1   .text
     _k_is_k_plus_j_l                      1A10      1E      30       2   .text
     _k_is_k_plus_j_i                      1A2E      12      18       3   .text
     _LSUB_k_is_k_minus_j                  1A40      1F      31       1   .text
     _LMUL_k_is_k_mul_j                    1A5F      81     129       2   .text
     _LDIVMOD                              1AE0      A4     164       3   .text
     _NEG_L_HX                             1B84      13      19       2   .text
     _ABS_L_HX                             1B97       6       6       2   .text
     _SPLITSIGN_L                          1B9D       A      10       1   .text
     _LMODU_k_is_k_mod_j                   1BA7       B      11       1   .text
     _LDIVU_k_is_k_div_j                   1BB2      17      23       2   .text
     _LDIVS_k_is_k_div_j                   1BC9      27      39       2   .text
     _LCMP_k_rel_j                         1BF0      1F      31       1   .text
     _LDEC                                 1C0F      1C      28       1   .text
     _LADD                                 1C2B       6       6       3   .text
     _LSUB                                 1C31       6       6       4   .text
     _LMUL                                 1C37       6       6       2   .text
     _LMUL_RC                              1C3D       6       6       1   .text
     _LDIVS_LC                             1C43       6       6       2   .text
```

```
     _LDIVS_RC                                 1C49       6       6       1   .text
     _LDIVU                                    1C4F       6       6       3   .text
     _LDIVU_RC                                 1C55       6       6       3   .text
     _LMODU_RC                                 1C5B       6       6       3   .text
     _LLSL                                     1C61      16      22       3   .text
     _LCMP                                     1C77       6       6       2   .text
     _POP32                                    1C7D      1A      26       5   .text
     initClock                                 1C97       F      15       1   .text
     initGenerator                             1CA6      13      19       1   .text
     setGenerator                              1CB9      59      89       1   .text
     initTimer                                 1D12      11      17       1   .text
     initInputCapture                          1D23       B      11       1   .text
     ISRToggleEdge                             1D2E      78     120       0   .text
     show                                      1DA6      E3     227       1   .text

*****************************************************************************************
UNUSED-OBJECTS SECTION
-----------------------------------------------------------------------------------------
NOT USED PROCEDURES
RTSHC08.C.o (ansiis.lib):
  _PUSH_ARGS_D _ENTER_UNARY_L64 _ENTER_UNARY_L64_4 _ENTER_BINARY_L64
  _ENTER_BINARY_L64_LC _ENTER_BINARY_L64_RC _IDIVMOD _SPLITSIGN
  _LAND_k_is_k_and_j _LOR_k_is_k_or_j _LXOR_k_is_k_xor_j _LMODS_k_is_k_mod_j
  _CMP24_k_rel_j _BMULS _BDIVS _BMODS _IMUL_STAR08 _IDIVS_STAR08 _IDIVU_STAR08
  _IMODS_STAR08 _IMODU_STAR08 _IDIVU_8 _IMODU_8 _IASR _ILSR _ILSL _ICMP _LINC
  _LNEG _LNOT _LADD_RC _LSUB_LC _LSUB_RC _LAND _LAND_RC _LOR _LOR_RC _LXOR
  _LXOR_RC _LDIVS _LDIVU_LC _LMODS _LMODS_LC _LMODS_RC _LMODU _LMODU_LC _LASR
  _LLSR _LCMP_RC _CMP24 _CMP24_RC _COPY _COPY_L _POP64 _STORE32 _STORE64
  _SEXT8_32 _SEXT16_32 _CALL_STAR08 _CALL_STAR08_FAR _Jump_Table_Addr
  _Jump_Table_Offset _Jump_Table_Header_Addr _Jump_Table_Header_Offset
  _Search_Table_16_Addr _Search_Table_16_Offset _Search_Table_8_Addr
  _Search_Table_8_Offset _PUSH_CC _POP_CC _CONV_FAR_TO_NEAR _CONV_FAR_TO_LINEAR
  _CONV_LINEAR_TO_FAR
Start08.c.o:
  errISR_TPM2CH0
generator.c.o:
  ISR_TPM1CH0
NOT USED VARIABLES
RTSHC08.C.o (ansiis.lib):
  _PowOfTwo_8 _PowOfTwo_16 _PowOfTwo_32 errno

*****************************************************************************************
COPYDOWN SECTION
-----------------------------------------------------------------------------------------
------- ROM-ADDRESS: 0x1E89 ---- SIZE       4 ---
Filling bytes inserted
 00020102
------- ROM-ADDRESS: 0x1E8D ---- RAM-ADDRESS: 0x102 ---- SIZE       2 ---
Name of initialized Object : CyclesPerSecond:2
 9897
------- ROM-ADDRESS: 0x1E8F ---- SIZE       2 ---
Filling bytes inserted
 0000

*****************************************************************************************
OBJECT-DEPENDENCIES SECTION
-----------------------------------------------------------------------------------------
Init                      USES _startupData loadByte
_Startup                  USES __SEG_END_SSTACK Init main
initPorts                 USES _PTFDD _PTBDD _PTADD _PTFD _PTBD _PTAD
main                      USES initClock initPorts initGenerator initTimer
                               initInputCapture show
_ENTER_BINARY_L           USES _PUSH_ARGS_L
_ENTER_BINARY_L_RC        USES _PUSH_ARGS_L
_ENTER_BINARY_L_LC        USES _PUSH_ARGS_L
_LMUL_k_is_k_mul_j        USES _k_is_k_plus_j_l _k_is_k_plus_j_i
_ABS_L_HX                 USES _NEG_L_HX
_SPLITSIGN_L              USES _ABS_L_HX
_LMODU_k_is_k_mod_j       USES _LDIVMOD
_LDIVU_k_is_k_div_j       USES _LDIVMOD
_LDIVS_k_is_k_div_j       USES _SPLITSIGN_L _LDIVMOD _NEG_L_HX
_LDEC                     USES _ENTER_UNARY_L
```

```
_LADD                     USES _ENTER_BINARY_L _LADD_k_is_k_plus_j
_LSUB                     USES _ENTER_BINARY_L _LSUB_k_is_k_minus_j
_LMUL                     USES _ENTER_BINARY_L _LMUL_k_is_k_mul_j
_LMUL_RC                  USES _ENTER_BINARY_L_RC _LMUL_k_is_k_mul_j
_LDIVS_LC                 USES _ENTER_BINARY_L_LC _LDIVS_k_is_k_div_j
_LDIVS_RC                 USES _ENTER_BINARY_L_RC _LDIVS_k_is_k_div_j
_LDIVU                    USES _ENTER_BINARY_L _LDIVU_k_is_k_div_j
_LDIVU_RC                 USES _ENTER_BINARY_L_RC _LDIVU_k_is_k_div_j
_LMODU_RC                 USES _ENTER_BINARY_L_RC _LMODU_k_is_k_mod_j
_LLSL                     USES _ENTER_UNARY_L
_LCMP                     USES _ENTER_BINARY_L _LCMP_k_rel_j
initClock                 USES _SOPT _ICGC1 _ICGC2 _ICGS1
initGenerator             USES _TPM1SC _TPM1C0SC setGenerator
setGenerator              USES _LDIVS_LC _TPM1MOD _LDEC _LMUL _LDIVS_RC
                                _TPM1C0V
initTimer                 USES _TPM2SC _TPM2MOD _TPM2CNT
initInputCapture          USES _TPM2C0SC
ISRToggleEdge             USES _PTDD _TPM2C0V TimeLowCycle TimeHighCycle _LCMP
                                _LSUB HighDifference LowDifference _POP32 _TPM2C0SC
show                      USES CyclesPerSecond HighDifference LowDifference
                                _LADD _LDIVU _POP32 TimeMicroSecond _LMUL _LMUL_RC
                                DutyCycle _PTFD _LDIVU_RC _LLSL _LMODU_RC _PTBD _PTAD

*****************************************************************************************
DEPENDENCY TREE
*****************************************************************************************
 main and _Startup Group
 |
 +- main
 |  |
 |  +- initClock
 |  |
 |  +- initPorts
 |  |
 |  +- initGenerator
 |  |  |
 |  |  +- setGenerator
 |  |     |
 |  |     +- _LDIVS_LC
 |  |     |  |
 |  |     |  +- _ENTER_BINARY_L_LC
 |  |     |  |  |
 |  |     |  |  +- _PUSH_ARGS_L
 |  |     |  |
 |  |     |  +- _LDIVS_k_is_k_div_j
 |  |     |     |
 |  |     |     +- _SPLITSIGN_L
 |  |     |     |  |
 |  |     |     |  +- _ABS_L_HX
 |  |     |     |     |
 |  |     |     |     +- _NEG_L_HX
 |  |     |     |
 |  |     |     +- _LDIVMOD
 |  |     |     |
 |  |     |     +- _NEG_L_HX            (see above)
 |  |     |
 |  |     +- _LDEC
 |  |     |  |
 |  |     |  +- _ENTER_UNARY_L
 |  |     |
 |  |     +- _LMUL
 |  |     |  |
 |  |     |  +- _ENTER_BINARY_L
 |  |     |  |  |
 |  |     |  |  +- _PUSH_ARGS_L         (see above)
 |  |     |  |
 |  |     |  +- _LMUL_k_is_k_mul_j
 |  |     |     |
 |  |     |     +- _k_is_k_plus_j_l
 |  |     |     |
 |  |     |     +- _k_is_k_plus_j_i
 |  |     |
```

```
|  |     +- _LDIVS_RC
|  |        |
|  |        +- _ENTER_BINARY_L_RC
|  |        |  |
|  |        |  +- _PUSH_ARGS_L         (see above)
|  |        |
|  |        +- _LDIVS_k_is_k_div_j  (see above)
|  |
|  +- initTimer
|  |
|  +- initInputCapture
|  |
|  +- show
|     |
|     +- _LADD
|     |  |
|     |  +- _ENTER_BINARY_L      (see above)
|     |  |
|     |  +- _LADD_k_is_k_plus_j
|     |
|     +- _LDIVU
|     |  |
|     |  +- _ENTER_BINARY_L      (see above)
|     |  |
|     |  +- _LDIVU_k_is_k_div_j
|     |     |
|     |     +- _LDIVMOD             (see above)
|     |
|     +- _POP32
|     |
|     +- _LMUL                (see above)
|     |
|     +- _LMUL_RC
|     |  |
|     |  +- _ENTER_BINARY_L_RC   (see above)
|     |  |
|     |  +- _LMUL_k_is_k_mul_j   (see above)
|     |
|     +- _LDIVU_RC
|     |  |
|     |  +- _ENTER_BINARY_L_RC   (see above)
|     |  |
|     |  +- _LDIVU_k_is_k_div_j  (see above)
|     |
|     +- _LLSL
|     |  |
|     |  +- _ENTER_UNARY_L       (see above)
|     |
|     +- _LMODU_RC
|        |
|        +- _ENTER_BINARY_L_RC   (see above)
|        |
|        +- _LMODU_k_is_k_mod_j
|           |
|           +- _LDIVMOD             (see above)
|
+- _Startup
   |
   +- Init
   |  |
   |  +- loadByte
   |
   +- main                 (see above)

ISRToggleEdge
|
+- _LCMP
|  |
|  +- _ENTER_BINARY_L      (see above)
|  |
|  +- _LCMP_k_rel_j
|
```

```
 +- _LSUB
 |  |
 |  +- _ENTER_BINARY_L      (see above)
 |  |
 |  +- _LSUB_k_is_k_minus_j
 |
 +- _POP32               (see above)

*******************************************************************************************
STATISTIC SECTION
-------------------------------------------------------------------------------------------

ExeFile:
--------
Number of blocks to be downloaded: 4
Total size of all blocks to be downloaded: 1689
```