Dokumentation Praktikum 01 GS

Sieb des Erasthoteles/Primzahlensieb

QUELLCODE: main.c

/\* Includes ------------------------------------------------------------------\*/

#include "main.h"

#include <stdio.h>

#include <string.h>

#include "TI\_Lib.h"

#include "tft.h"

#include "stm32f4xx.h"

#include "stm32f4xx\_gpio.h"

#include "stm32f4xx\_rcc.h"

/\* Private typedef -----------------------------------------------------------\*/

/\* Private define ------------------------------------------------------------\*/

/\* Private macro -------------------------------------------------------------\*/

/\* Private variables ---------------------------------------------------------\*/

/\* Private function prototypes -----------------------------------------------\*/

/\* Private functions ---------------------------------------------------------\*/

/\*\*

\* @brief Main program

\* @param None

\* @retval None

\*/

int prim\_anz;

int start;

int ende;

int main(){

start = 40;

ende = 50;

Init\_TI\_Board();

TFT\_cls();

init\_prim\_arr();

calc\_prim\_arr();

// print\_prim\_arr();

prim\_anz = calc\_sum\_prim(start,ende);

print\_ergebnis\_TFT();

return 0;

}

Sieb.c

#include <math.h>

#include <stdio.h>

#include "main.h"

// ---------------------------------------------------------------- ALGORITHM -

// -------------------------------------------------- INITIALISATION ----------

char prim\_arr**[**ANZ**];**

void init\_prim\_arr**(){**

// indizes

int i**;**

// -- DEFINITION --

prim\_arr**[**0**]=**0**;**

prim\_arr**[**1**]=**0**;**

**for** **(**i**=**2**;**i**<**ANZ**;**i**++){**

prim\_arr**[**i**]=**1**;**

**}**

**return;**

**}**

// ------------------------------------------------------ CALCULATION ---------

void calc\_prim\_arr**(){**

// -- DEKLARATION --

// indizes

int i**,** endi**,** j**;**

// -- DEFINITION --

endi**=(**int**)**sqrt**((**double**)**ANZ**);**

// -- CALCULATION --

**for** **(**i**=**0**;**i**<**endi**;**i**++){**

// printf("%i ",i);

**if** **(**prim\_arr**[**i**]==**1**)** **{**

**for** **(**j**=**i**+**i**;**j**<**ANZ**;**j**=**j**+**i**)** **{**

prim\_arr[j]=0;

}

}

}

return;

}

// ----------------------------------------------------- SUMMARYSE ------------

int calc\_sum\_prim(int start, int ende){

int i;

int anz\_prim;

if (ende > ANZ) {

ende = ANZ;

}

anz\_prim=0;

for (i=start;i<ende;i++){

if(prim\_arr[i]==1){

anz\_prim++;

}

}

return anz\_prim;

}

// ---------------------------------------------------------------------- EOF -

Ausg.c

#include "main.h"

#include <stdio.h>

#include <string.h>

//#include "TI\_Lib.h"

#include "tft.h"

//int anz\_prim;

void print\_ergebnis\_TFT**(){**

char ausgabe**[**80**];**

// if (!oefters\_ausgefuehrt){

// oefters\_ausgefuehrt = true;

// }

sprintf**(**ausgabe**,** "Primzahlenanzahl zwischen %d und %d: %d \n"**,** start**,** ende**,** prim\_anz **);**

TFT\_puts**(**ausgabe**);**

**return;**

**}**

Main.h(sieb.h)

/\* Define to prevent recursive inclusion -------------------------------------\*/

#ifndef \_\_MAIN\_H

#define \_\_MAIN\_H

#ifdef \_\_cplusplus

extern "C" **{**

#endif

#define FULL\_ASSERT 1

#define USE\_FULL\_ASSERT 1

/\* Includes ------------------------------------------------------------------\*/

#include "stm32f4xx.h"

/\* Exported typedef -----------------------------------------------------------\*/

// #define countof(a) (sizeof(a) / sizeof(\*(a)))

// typedef enum {FAILED = 0, PASSED = !FAILED} TestStatus;

// --------------------------------------------------------------- KONSTANTEN -

#define ANZ 100

// --------------------------------------------------------------- PROTOTYPEN -

void init\_prim\_arr**(**void**);**

void calc\_prim\_arr**(**void**);**

int calc\_sum\_prim**(**int**,** int**);**

void print\_ergebnis\_TFT**(**void**);**

// -------------------------------------------------------------- DEKLARATION -

extern char prim\_arr**[**ANZ**];**

extern int prim\_anz**;**

extern int start**;**

extern int ende**;**

#ifdef \_\_cplusplus

**}**

#endif

#endif /\* \_\_MAIN\_H \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* (C) COPYRIGHT STMicroelectronics \*\*\*\*\*END OF FILE\*\*\*\*/

Struktur:

Das Projekt besteht aus 3 Modulen und 1ner Headerdatei.

sieb.c:

void init\_prim\_arr() // initialisiert das Array, auf dem gerechnet wird.

void calc\_prim\_arr() // berechnet die primzahlen (algo) im Array

int calc\_sum\_prim() // berechnet die Anzahl zwischen Start und einem Ende (beide global)

// dabei wird das Ende auf das obere Limit des Arrays gesetzt falls ende größer als dieses ist.

// unteres Limit (o) und Start wird nicht verglichen.

ausg.c:

static bool oefters\_ausgefuehrt // abfrage, weil gefordert, ob es die erste Ausgabe ist.

// wenn ja, dann wird der TFT im folgenden Unterprogramm entsprechend vorbereitet.

void print\_ergebnis\_TFT() // gibt auf dem TFT-Bildschirm die Anzahl der Primzahlen im intervall an

main.c:

void main() // ruf nacheinander die einzelnen Unterprogramme auf. Dabei wird die Variable

// prim\_anz mit calc\_prim\_arr() definiert. Danach wird es ausgegeben.

sieb.h (main.h)

Konstanten:

ANZ = länge des Arrays

Prototypen aller Unterprogramme

Globale Variablen:

int prim\_arr

int prim\_anz

int start

int ende

Der Algorithmus steht im Unterprogramm calc\_prim\_arr und ist das Sieb des Erathostenes:

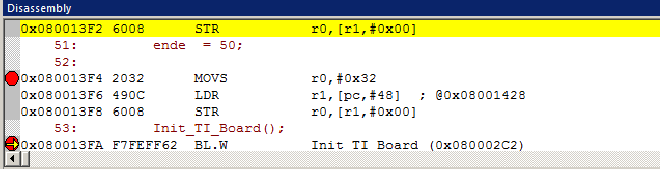
Dabei wird immer das vielfache der betrachteten Primzahl aus dem Kreis des potenziellen

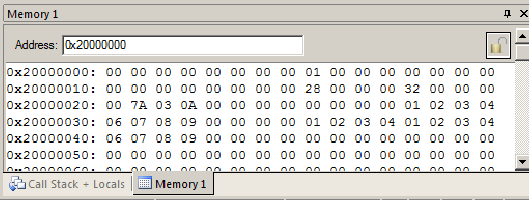
Primzahlen herausgesiebt. Dann wir die nächst höhere Primzahl betrachtet und die Vielfachen

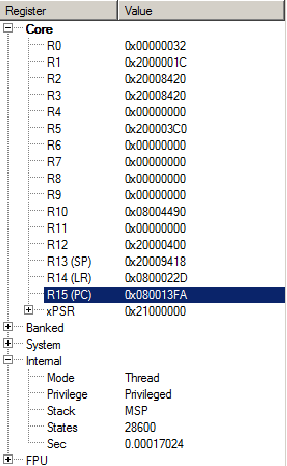
aus dem Kreis der potenziellen Primzahlen gelöscht usw.

Screenshots:

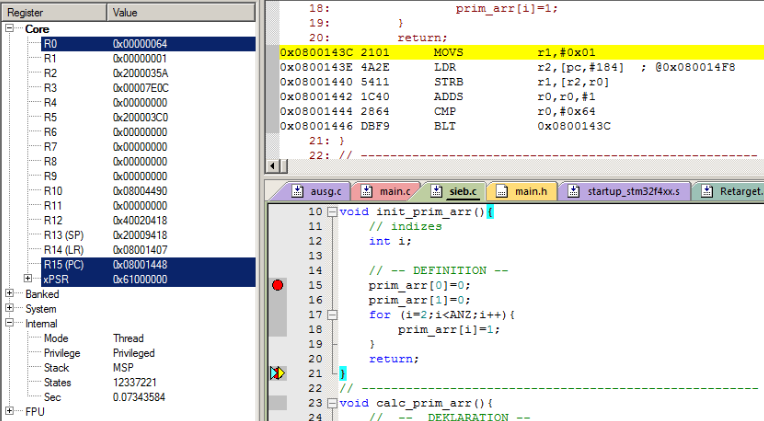
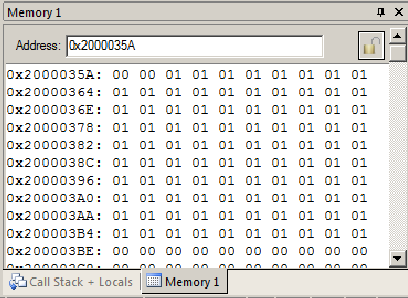
1. Definition von den Variablen „start“ und „ende“ (in unserem Fall 40 und 50). Im Memory Fenster 0x20000018 (start = 0x28= 40) und 0x2000001C (ende =0x32= 50). Hier werden die globale Variablen abgespeichert.

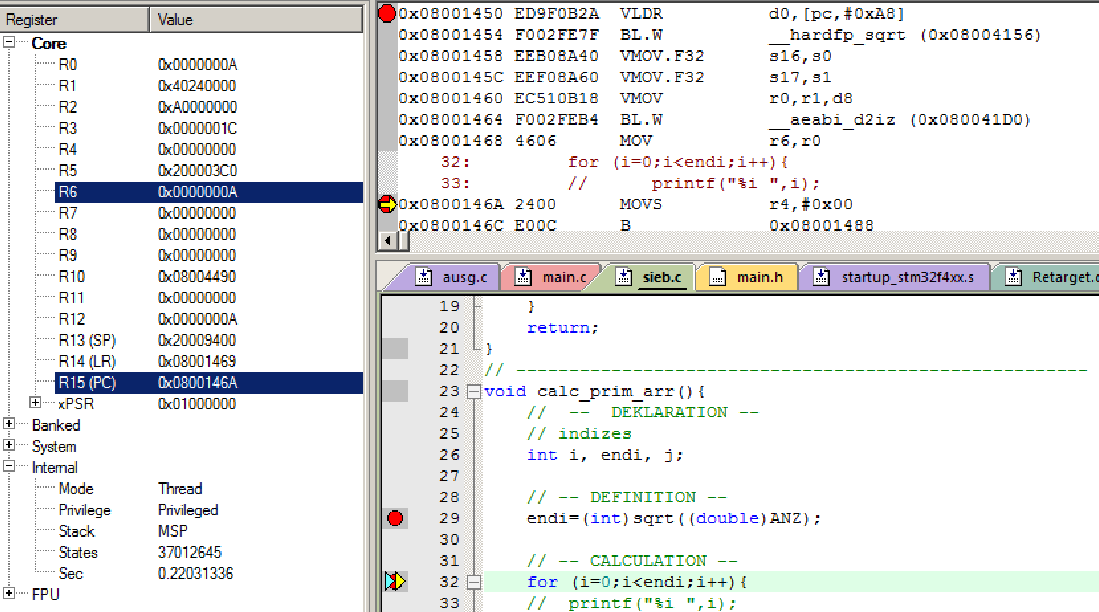
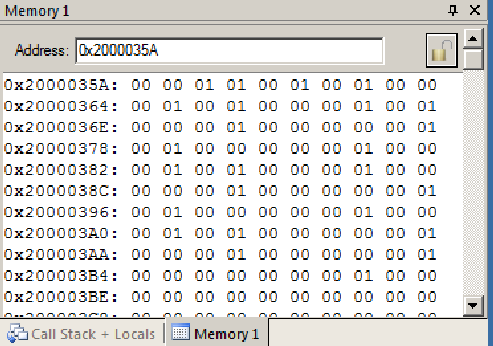
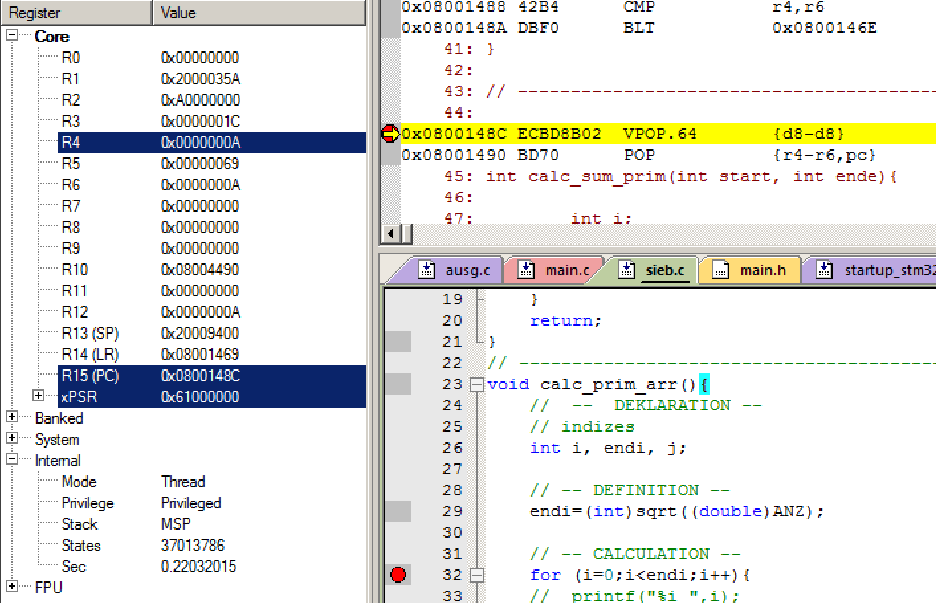


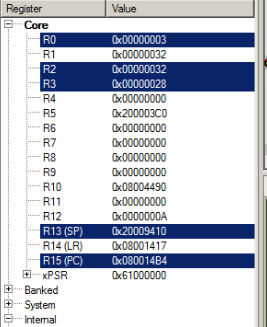




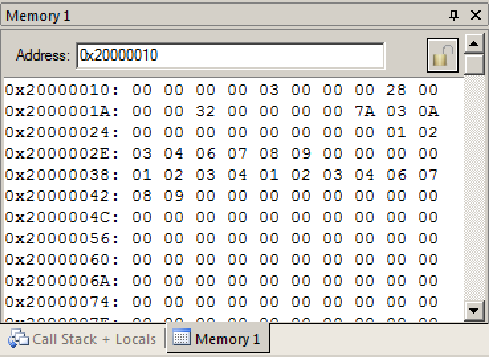
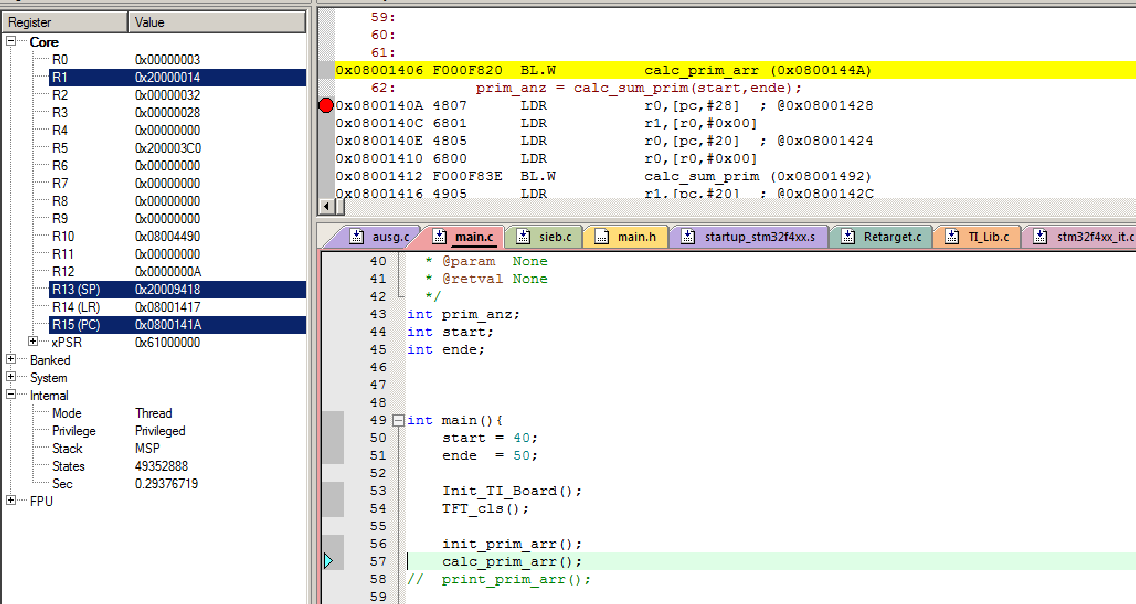
1. Defintion Primzahlen-Array in der Initialisierung. PrimzahlenArray geht ab 0x2000035A los bis …3BD (0…100). Hier werden erstmal alle mit 1 besetzt, außer [0], [1].



1. Lokale Variable für „endi“ (nur im Register R6)
2. Nach dem Durchlaufen der calc. („i“ ist in R4, aktuell der letzte mögliche Wert (endi)). Im Memory Fenster sehen wir die Primzahlen. (Zwischen 40 und 50 sind drei 1sen, zwischen 382 – 38C)
3. Lokale (Grenz-) Variablen (r0: Rückgabewert (Anzahl Primzahlen), r2 ist die obere Grenze (50>Hex32) und r3 die untere (40).



1. Ergebnis wird als globale Variable gespeichert (R0 -> prim\_anz, vor start und ende gespeichert, da die Deklarationsreihenfolge auch so war)-



1. Ausgabe, lokal

Deswegen laden wir bei 93C4… (Ascii Darstellung)

