ti23_assignment_13_Alabrsh_Panov_Zeitler

<table>
<tr><td>1a)</td><td>Explained Functions of high_lvl.cpp:

<table>
<tr><td>Fkt0</td><td>Gibt Eingabeparameter i_value als Rückgabewert zurück</td></tr>
<tr><td>Fkt1</td><td>Gibt immer den Wert 0 als Rückgabewert zurück</td></tr>
<tr><td>Fkt2</td><td>Wenn Eingabeparameter i_option kleiner als 32 ist, wird 1 zurückgegeben.
Ansonsten wird 0 zurückgegeben</td></tr>
<tr><td>Fkt3</td><td>Es gibt 2 Eingabeparameter (Pointer). Wenn der Wert an der Adresse des ersten Pointers kleiner als 25 ist, dann wird der Wert an der Adresse des zweiten Pointers auf 1 gesetzt.
Ansonsten wird der Wert auf 0 gesetzt.</td></tr>
<tr><td>Fkt4</td><td>Es gibt 3 Eingabeparameter (x, y, z).
Wenn x kleiner als y und x kleiner als z ist, dann wird der Wert 1 zurückgegeben.
Wenn y kleiner als z ist wird der Wert 2 zurückgegeben.
Ansonsten wird in den anderen Fällen der Wert 3 zurückgegeben.</td></tr>
<tr><td>Fkt5</td><td>Es gibt 2 Eingabeparameter, einen Wert und einen Pointer.
In der Schleife wird der Wert an der Adresse vom Pointer immer um 1 erhöht.
Die Schleife wird durchgeführt von 0 bis Anzahl an Iterationen.</td></tr>
<tr><td>Fkt6</td><td>Es gibt 3 Eingabeparameter, 2 Werte und einen Pointer.
-> Anzahl der Schleifendurchgänge (i_nIters)
-> den Wert, der in jedem Schleifendurchgang zum Zeigerwert addiert werden soll (i_inc)
-> einen Zeiger auf den zu verändernden Wert (io_value)
Schleife wird mindestens einmal durchlaufen, wobei in jedem Durchgang der Wert am Zeiger um den angegebenen Inkrementwert erhöht wird. Anzahl der Schleifendurchgänge wird durch eine temporäre Variable (l_va) repräsentiert, die dekrementiert wird, bis sie 0 erreicht.
Wenn die Anzahl der Schleifendurchgänge auf 0 gesetzt ist, wird die Schleife weiterhin ausgeführt, bis die temporäre Variable den maximalen Wert eines uint64_t erreicht, da keine Prüfung auf Null erfolgt, nachdem sie einmal dekrementiert wurde.</td></tr>
<tr><td>Fkt7</td><td>Es gibt 3 Eingabeparameter, einen Wert und 2 Pointer.
In der Schleife werden n Elemente (Werte) von dem einen Array (i_valuesIn) in das andere Array (i_valuesOut) kopiert.</td></tr>
</table>
</td></tr>
<tr><td>1b)</td><td>Implemented low_lvl Functions. → siehe src (low_lvl.h, low_lvl.s)</td></tr>
<tr><td>1c)</td><td>Added informative comments to assembly code: → siehe src (low_lvl.s)</td></tr>
<tr><td>1d)</td><td>Verified assembly code by extending driver such that outputs such that outputs the result of the low-level implementation are given next to the results of the high-level implementation.
→ siehe src (driver.cpp) → (driver.o)</td></tr>
<tr><td>2a)</td><td>Based on the provided C implementation, write both functions in assembly code:
→ siehe src (gcd.cpp, gcd_c.c, gcd_s.s)
Verified for corrcectness → scr (gcd.o)</td></tr>
<tr><td>2b)</td><td>Added informative comments to assembly code: → siehe src (gcd_s.s)</td></tr>
</table>

Aufgabenbearbeitung:

Aufgabe 1 → Christian, Cora, Rahaf
Aufgabe 2 → Christian, Cora, Rahaf