Programmeertechnieken 2017
Huiswerk 2: Pointer arithmetic & bit-wise operators

Deadline: Vrijdag 3 maart, voor het einde van de dag.

Inleveren

Er mag worden gewerkt in tweetallen. De wijze van inlevering is een verslag in TXT formaat (platte tekst, dus geen PDF, Word, ODT, etc.), waarin alle opgaven zijn uitgewerkt. Beargumenteer al uw antwoorden. Zorg ervoor dat namen en studentnummers duidelijk bovenaan het verslag zijn vermeld. Geef het verslag de volgende bestandsnaam:

`hw2-sXXXXXXX-sYYYYYYY.txt`

Vul op de plek van `XXXXXXX` en `YYYYYYY` de bijbehorende studentnummers in. De inzendingen kunnen worden verzonden per e-mail naar `pt2017 (at) handin.liacs.nl` met als onderwerp "PT Huiswerk 2".

Opgave 1

Beantwoord de onderstaande vragen. Beargumenteer uw antwoorden.

a) Gegeven `double *a = (double *)0x2000;`. Wat is de waarde van `a` na `a += 18`?

b) Gegeven `uint16_t *c = (uint16_t *)0x1000;`. Wat is de waarde van `c + 4 * sizeof(uint32_t)`?

c) Gegeven `float **b = (float **)0x400;`. Wat is de waarde van `b + 2 * 5`?

d) Gegeven `char *str = "hello world"; char **p = &str;`. Schrijf een expressie met `p` die het karakter "r" uit de string leest.

e) Gegeven `float A[] = { 3.14f, 4.54f, 9.54f, 0.34f };`. Maak de volgende expressie af om het geheugenadres waar `9.54f` staat te berekenen: `uintptr_t a = (uintptr_t)&A[0] ....`

f) Gegeven een void pointer `void *p`, welke wijst naar een locatie in het geheugen waar de volgende datastructuur staat:

Byte offset	Field type
0	`uint32_t`
7	`uint8_t`
16	`double`

Schrijf de benodigde pointer arithmetic om de velden van deze datastructuur in aparte variabelen te laden.

Opgave 2

Stel we hebben een 2-dimensionale array van `double`s bestaande uit 256 rijen en 64 kolommen. De array is opgeslagen door de *kolommen* achter elkaar in het geheugen te plaatsen. Maak de volgende functie af, die gegeven een pointer naar het begin van de array en rij- en kolomaanduiding het juiste element uit het geheugen laadt. Geef ook een toelichting op de gegeven uitwerking (mag ook als commentaar in de gegeven functie).

```
#define N_ROWS 256
#define N_COLUMNS 64

static inline double get_element(const double *A,
                                 const int     row,
                                 const int     column)
{
   ......
}
```

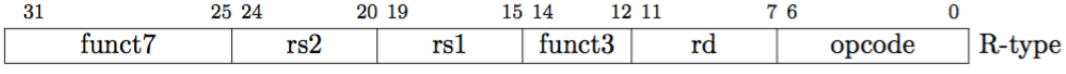
Opgave 3

Als Embedded Software Engineer werkt u aan een drukmachine welke kan omgaan met een speciale 26-bits kleurcodering. Ook al hebben we maar 26 bits nodig, we gebruiken een `uint32_t` om een kleur op te slaan. De kleur bestaat uit vier componenten: C, M, Y en K (Cyan, Magenta, Yellow, Key). Voor elk C, M en Y zijn 7 bits gereserveerd. Voor K 5 bits. Component K is opgeslagen op bits 0 t/m 4, Y op 5 t/m 11, M op 12 t/m 18 en C op 19 t/m 25.
Het is uw taak om functies te schrijven die dergelijke kleuren kunnen manipuleren. Maak uiteraard gebruik van bitwise operaties en houd de code zo kort mogelijk. De kleur wordt steeds als parameter doorgegeven of verwacht als returnwaarde, type `uint32_t`. Schrijf de volgende functies:

```
/* opgave4.cc */

#include <cstdio>
#include <cstdint>

enum Component { C = 3, M = 2, Y = 1, K = 0 };

/* Gegeven waarden voor de verschillende componenten als parameters, construeer
 * en return de 26-bit kleur.
 */
uint32_t make_color(uint8_t C_val, uint8_t M_val, uint8_t Y_val, uint8_t K_val)
{ }

/* Wis alle kleur componenten (C, M en Y) in "color". */
void clear_colors(uint32_t &color)
{ }

/* Return "true" als "color" een grijstint is (anders false); dat wil zeggen
 * alleen component "K" wordt gebruikt en C, M en Y zijn nul.
 */
bool is_grayscale(const uint32_t &color)
{ }

/* Return de waarde van het gevraagde component uit "color". */
uint8_t get_component(const uint32_t &color, const Component component)
{ }

/* Zet de waarde van het gevraagde component in "color" op de waarde "value".
 * Vergeet niet eerst de bestaande waarde van het component te wissen!
 */
void set_component(uint32_t &color, const Component component, uint8_t value)
{ }
```

Opgave 4

In de RISC-V architectuur hebben alle CPU instructies een lengte van 32 bits, waarin alle benodigde gegevens voor een instructie zijn geëncodeerd. We bekijken in deze opgave een zogenaamde "R-type" instructie, welke een operatie toepast op twee registers en het resultaat wegschrijft in een derde register: $R_d \leftarrow R_{s1}(\text{op})R_{s2}$. Een geëncodeerde R-type instructie bestaat uit zes velden en ziet er als volgt uit:



Bron: The RISC-V Instruction Set Manual. Volume I: User-Level ISA. Version 2.0

Stel nu er is een variabele `uint32_t instr` gegeven, welke een dergelijke instructie bevat. Geef een C-code die alle velden van de instructie isoleert in aparte variabelen, door gebruik te maken van bitwise operators. Bijvoorbeeld `rs1` is een 5-bit veld dat een registernummer uit het interval $[0, 31]$ bevat. Zorg dus dat de waarde van het veld `rs1` wordt opgeslagen als een integer tussen 0 en 31 in een variabele `rs1`.