0.1 Discussion

This section contains a debate about the goals of this project, and if they has been met. There is also a section which contains reflections about the project.

0.1.1 Characteristic of SPLAD

this section compares the design criteria set forward in section **??**, and checks if these criteria has actually been met.

"Simplicity" was set to be high for SPLAD. This, as explained in section **??** has been achieved.

0.1.2 Simplicity of SPALD compared with C

This section will compare pieces of code written in SPLAD, with pieces of code written in C. The first code example, seen on listring 1, is an example of a simple SPLAD program, which prints out "HelloWorld" to the LCD.

```
1  function pour return nothing using(int a, double b)
2  begin
3    return nothing;
4  end
5
6  function RFIDFound return nothing using(int a, int b)
7  begin
8    string message <-- "Hello World!";
9    /* Print message on line 1 on LCD */
10   call LCDPrint(message, 1);
11
12   return nothing;
13 end
```

Listing 1: Hello world program in SPLAD

Line 1-4 of 1 contains the pour function, which is required by the SPLAD compiler, but is not relevant in this small example. The string "message" is declared on line 7, which is what will be printed to the LCD. On line 9, the function LCDPrint, which is provided by the SPLAD compiler is called, and the message is printed to the display. The message will be printed when an RFID-tag is found by the RFID-reader.

A hello world program written in Arudino C/C++, can be seen on listing 2. It should be noted that this is an example provided by the Arduino IDE [**?**]. Line 2 of this example includes the LCD Library. On line 5 the LCD display must be initialized with the pins on the Arduino. On line 9, the LCD display is setup, which means that the appropriate number of columns and rows is set depending on the particular model used. At last on line 11, the message is printed to the LCD.

```
1 // include the library code:
2 #include <LiquidCrystal.h>
3
4 // initialize the library with the numbers of the interface pins
5 LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
6
7 void setup() {
8   // set up the LCD's number of columns and rows:
9   lcd.begin(16, 2);
10  char message[ ] = "Hello World";
11  // Print a message to the LCD.
12  lcd.print(message);
13 }
14
15 void loop() {
16
17 }
```

Listing 2: Hello world program in Arduino C/C++

The difference between the SPLAD program, and the Arduino program is clear: The SPLAD program is much more specialized with the target being drinks machine, which means that there is an LCD print function provided by default. This is not the case in a normal Arduino program, because the Arduino is aimed much more at general purpose. This means that users of the SPALD language need not think about which pins the LCD is connected to while writing programs. The Arduino does not provide a string-type, which means that strings are implemented by char arrays. In the SPLAD language there is a string type, which might seem more intuitive for novice programmers. The assignment is a bit different in SPLAD compared to C-notation. In SPLAD an assignment is denoted by '<–', which makes it completely clear what is assigned to what. In C, the assignment is denoted by '=', which might confuse novice programmers, because '=' generally is used to denote equality in for example mathematics. Also when using the '=', it is not really clear what is assigned to what.

0.1.3 Other Criteria

"Orthogonality" was set to be medium for our language. Since it is not possible to make classes or constructs in our language. But it is possible to call a function in an expression. Therefore we think that we have achieved this characteristic at the level we what it to be.

Fatal: kan ikke
rigtige bøjning

"Data types" was set to be medium for our language. Because we only have five primitive types and two special types, our language have many types, but still not that many types which makes it easily to have an overview them, and it is not possible to make classes or constructs. Thereby this characteristic have been achieved.

Fatal: kan ikke
rigtige bøjning

"Syntax design" was set to be high for our language. Since most of our language have been made in the way to be better understood what a "for loop" do and where block start and ends. Therefore have this characteristic been achieved in our programming language.

Fatal: kan ikke
rigtige bøjning

"Support for abstraction" was set to be medium for our language. Because our language is an abstraction in itself. There is particular form for abstraction in our language. But in our language it is not possible to do abstraction. This limit in our language is for helping beginners to get a easier and faster understanding of a given code in our language.

"Expressivity" was set to be low for our language. Since our language had it focus to be as simple as possible, are expressivity not done in our language.

"Type checking" was set to be high for our language. To make sure that all type errors could be find is fund when a program is compiled. Therefore a lot of work and tests have been into the type checker. There were a some problems, like the build-in functions that a Arduino board have. This make it hard to make a type checker that are will find all errors. But all the types, expressions, parameters that are in our language are type check and therefore have this