DSPJockey

Brian Bourn, Abhinav Mishra
Addisu Petros, Vanshil Shah

COMS 4115 Programming Languages & Translators
Prof. Stephen Edwards
December 17, 2014
Motivation

• Digital Signal Processing used in fields of Electrical Engineering, Audio mixing, and even algorithmic trading
• Many useful operations that can be done in signal processing such as convolution, filtering, time shifting
• Lack of tools to build and manipulate signals easily
• Notion of global time for a signal only apparent in languages that model hardware such as SystemC
Why DSPJockey?

- Provides a simple framework for creating and manipulating signals using Signal data type
- C-like syntax including primitive data types
- Includes built in functions common in DSP
- Global time for each signal: easy to access signal at current time or at a previous time (past)
Language Tutorial

- DSPJockey uses C/C++ like syntax
- Includes the primitive data types, int, float, string, and bool
- Aggregate data types are Array and Signal
- Functions must have a return type
Arrays are similar to C as they are lists that are of a fixed size and contain float values.

To create and initialize the array of a given size, say 10
let arr = Array[10];

To access the third element in this array
float x = arr[2];
Signal

Signals are similar to arrays are implemented as a circular buffer and its values are accessed by using the time keyword.

To create a signal:

```javascript
let sig = new Signal[];
```

To access the value of signal at current time:

```javascript
float y = sig[time];
```

The value at a previous time can be accessed by subtracting the number of time units from time:

If we want to access the value at 2 time units before current time

```javascript
float z = sig[time-2];
```
When an operation is performed on a signal, it is done over the whole signal.

Example:

\[ \text{sig}[\text{time}] = \text{sig}[\text{time}] + 1 \]

will increment all the samples in the signal by one.
Control Flow

- If/else, while and for loops follow the same exact syntax as C.
- If/else statements are exactly similar to C and the else statement is not required.
  
  ```
  if ( boolean_condition ) {
    
  }
  else {
    
  }
  ```

- While loop:
  ```
  while ( boolean_condition ) {
    
  }
  ```

- For loops:
  ```
  for(initialization; boolean_condition; iteration_step){
    
  }
  ```
Functions

- Functions are similar to C/C++ but there are two types of functions,
  - 1. normal functions, return a primitive type
     ```
     int x(args){
     }
     ```
  - 2. stream functions used for manipulating signals
     ```
     stream x(args){
     }
     ```
- Every single .dj file must contain a main function.
- Calling a function is done in the same way as C/C++
  ```
  int result = function(float a);
  ```
Built-in Functions

• The print is just used for printing to standard output
  print “hello world”;
  print 5;
• The Sum function takes in a id, starting index, ending index and expression and evaluates the summation
  sum x = 1 to 2:x+1;//5
Language Implementation

.\dj file \rightarrow Scanner \rightarrow Parser

Code generation \leftarrow Semantic Analyzer/SAST \leftarrow AST

Generated C++ Code
Lessons Learned

• Start on time!
• Understand components of compiler before beginning
• Develop in smaller chunks
• Learn Ocaml before or right at the beginning of the course
• Think about how all the components connect so that you don’t have to end up going back to previous sections
DEMO!!!
Any Questions???