Script generated by TTT

Title: Petter: Virtual Machines (30.04.2019)

Date: Tue Apr 30 10:16:13 CEST 2019

Duration: 86:24 min

Pages: 4

For ease of comprehension, we use symbolic jump targets. They will later be replaced by absolute addresses.

Instead of absolute code addresses, one could generate relative addresses, i.e., relative to the actual PC.

Advantages

- smaller addresses suffice most of the time;
- the code becomes relocatable, i.e., can be moved around in memory.

Remark

- The jump table could be placed directly after the code for the Macro check. This would save a few unconditional jumps. However, it may require to search the **switch**-statement twice.
- If the table starts with u instead of 0, we have to decrease the R-value of e by u before using it as an index.
- If all potential values of e are definitely in the interval $[0, k]$, the macro check is not needed.

$$
\begin{array}{lcl}
\text{code}_L\ e_1[e_2]\ \rho & = & \text{code}_R\ e_1\ \rho \\
 & & \text{code}_R\ e_2\ \rho \\
 & & \text{loadc}\ |t| \\
 & & \text{mul} \\
 & & \text{add}
\end{array}
$$

Remark

- In C, an array is a pointer. A declared array a is a pointer-constant, whose R-value is the start address of the array.
- Formally, we define for an array e: $\text{code}_R\ e\ \rho = \text{code}_L\ e\ \rho$
- In C, the following are equivalent (as L-values):

$$2[a] \qquad a[2] \qquad *(a+2)$$

Normalization: Array names and expressions evaluating to arrays occur in front of index brackets, index expressions inside the index brackets.

5.2 Structures

In Modula and Pascal, structures are called Records.

Approach

We manage a separate environment ρ_{st} for each structure type st.

Be **struct** $st\ x$; part of a declaration list where

struct st { **int** a; **int** b; };

- x has as relative address the address of the first cell allocated for the structure.
- The components have addresses relative to the start address of the structure st. In the example, these are given by $\rho_{st} = \{a \mapsto 0, b \mapsto 1\}$.