Lehrstuhl für Software and Computational Systems
Prof. Dr. Gidon Ernst, Marian Lingsch-Rosenfeld

Übungen zur Vorlesung
Formale Spezifikation und Verifikation

Wintersemester 2024/25

Übungsblatt 03

Bekanntgabe am 11.11.2024

1 Symbolische Suche

Für diese Aufgabe wird die Lösung für Programm 1a in der Übung vorgerechnet, bitte bereiten Sie die Aufgabe für Programm 1b selbständig vor.

Gegeben sind zwei C-Programme 1a und 1b auf der nächsten Seite. Als Spezifikation gilt, dass das `ERROR`-Label nicht erreicht werden darf. Sollte dies dennoch der Fall sein, so gilt die Spezifikation als verletzt bzw. nicht erfüllbar.

Aufgaben:

1. Modellieren Sie die beiden C-Programme als Kontrollflussautomat. Markieren Sie die Knoten jeweils mit den Zeilennummern.
2. Verifizieren Sie die Programme mit Hilfe der symbolischen Suche. Geben Sie die Menge der erreichbaren Zustände als Formel über den Programmvariablen sowie dem Programmzähler `pc` an.
3. Falls das Programm die Spezifikation verletzt: Geben Sie eine Belegung der Variablen der Formel an, die diese Verletzung beschreibt.

Hinweise:

- Für die externe Funktion `__VERIFIER_nondet_uint()` können Sie annehmen, dass sie bei jedem Aufruf einen nichtdeterministischen Wert aus dem Wertebereich des Typs `unsigned int` zurückgegeben wird. Das umfasst alle Zahlen aus dem Bereich 0 bis $2^{32} - 1$.

```c
extern unsigned int
    __VERIFIER_nondet_uint();

unsigned int x = 0;
unsigned int y = 0;

int main(void) {

  x = __VERIFIER_nondet_uint();
  y = __VERIFIER_nondet_uint();

  if (x != 0) {
    y = 0;
  }
  if (y != 0) {
    x = 0;
  }

  if (x * y == 0) {
    ERROR:
    return 1;
  }
  return 0;
}
```

(a)

```c
extern unsigned int
    __VERIFIER_nondet_uint();

unsigned int x = 0;
unsigned int y = 0;

int main(void) {

  x = __VERIFIER_nondet_uint();
  y = __VERIFIER_nondet_uint();

  if (x * y == 0) {
    return 0;
  }

  if (x * y == 0) {
    ERROR:
    return 1;
  }
  return 0;
}
```

(b)