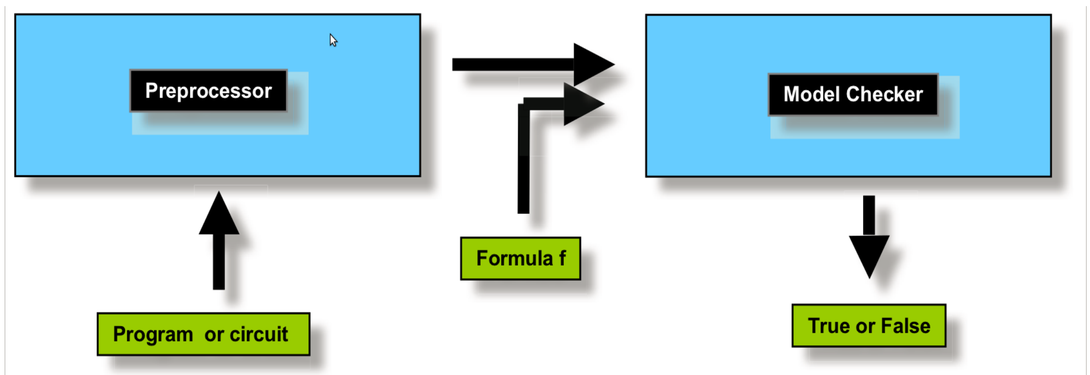
Multicore DSP Software Verification

SAT-Based Model Checking

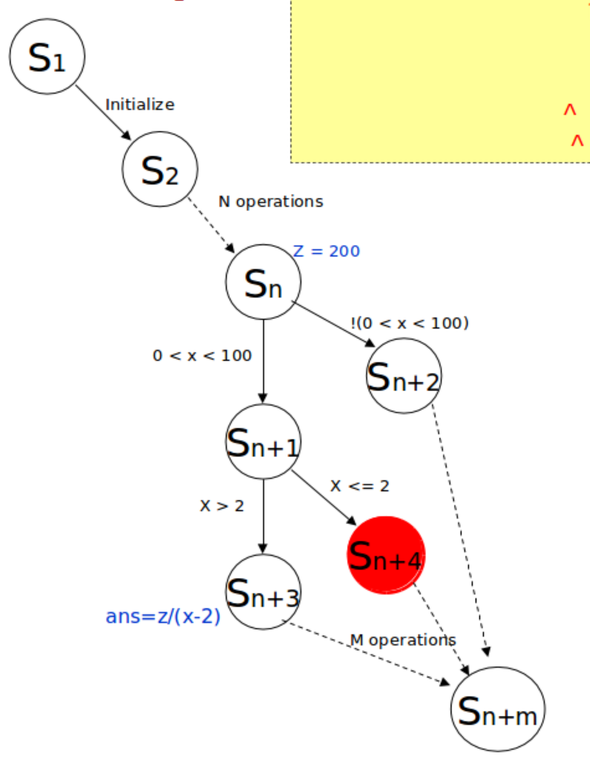


- **SAT solver** as back-end
- Can handle
  real-world languages
  like C, C++, Java
- **Bounded MC** for bug finding
- **Full MC** for verification

Example:

Automatically check if this can be true?

```
z = 200
^ 0<x<100
^ x <= 2
```



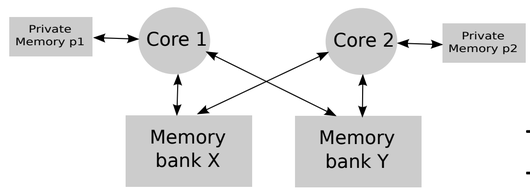
- Main tool used by us: **CBMC**

Ashok Kelur
ashok.kelur.1807@student.uu.se
Dr. Philipp Rümmer
philipp.ruemmer@it.uu.se

UPPSALA UNIVERSITET
Department of Information Technology

DSP-C Extensions

- **Fixed-point** data types:
  `__fixed` and `__accum`
- **Memory labels** for
  divided memory space




```
__X  int *shared_int;
__Y  int  buf[100];
__p1 int *private_int;
```

- Dedicated **register sets**

```
__reg_mul_s1 = X;
__reg_mul_s2 = y;
mul_64_fixed
res = __reg_128_r;
```

- **Circular buffers**

```
__X  buf[10];
__Y  my_data[100];

for(i=0; i<100; i++)
    my_data[i] = buf[i];
```

- Most features are supported by
  extended version of CBMC

Real-World Case Studies

- **Sequential challenges**
  - Dynamic system configuration
  - Initialisation process
  - Large code bases
  - Hardware accelerators
  - Assembler code
  - Highly incomplete specs
  - *And more ...*
- **Concurrent/multicore challenges**
  - Race conditions
  - Deadlocks
  - Verif. of memory management
  - Verif. of use of shared resources
- **Large-scale verification**
  case study in industrial context