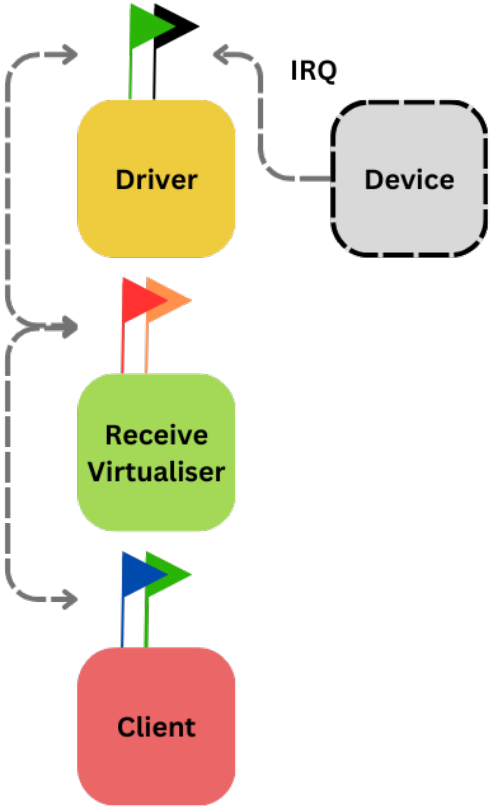
Using Model Checking to Develop and Verify sDDF Communication Protocols

Courtney Darville

seL4 Device Driver Framework

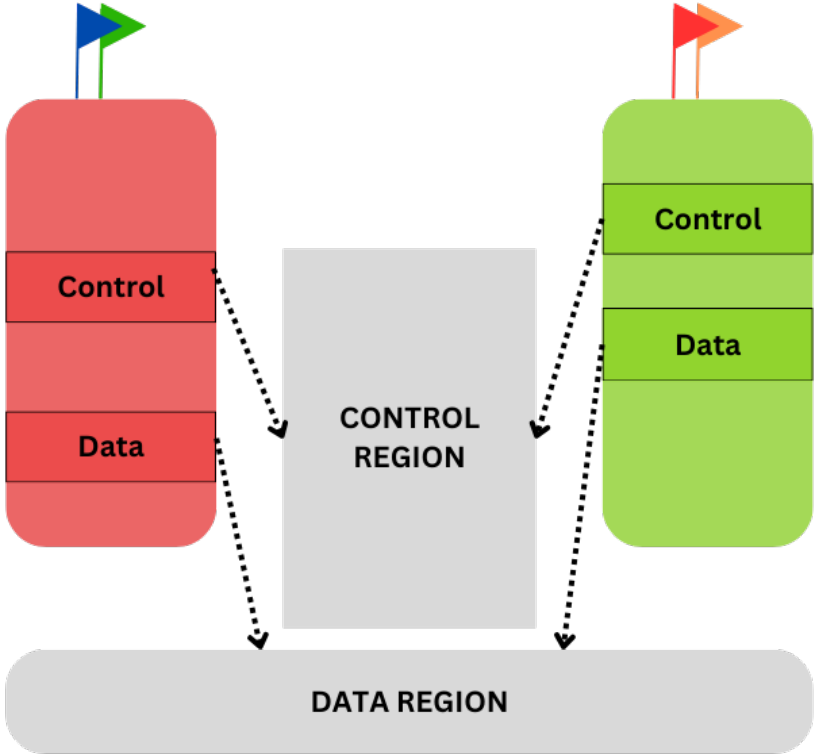
- Device drivers and supporting components
- Running natively on seL4
- Interface specifications

➡ *Principle of separation of concerns*



sDDF Component Communication

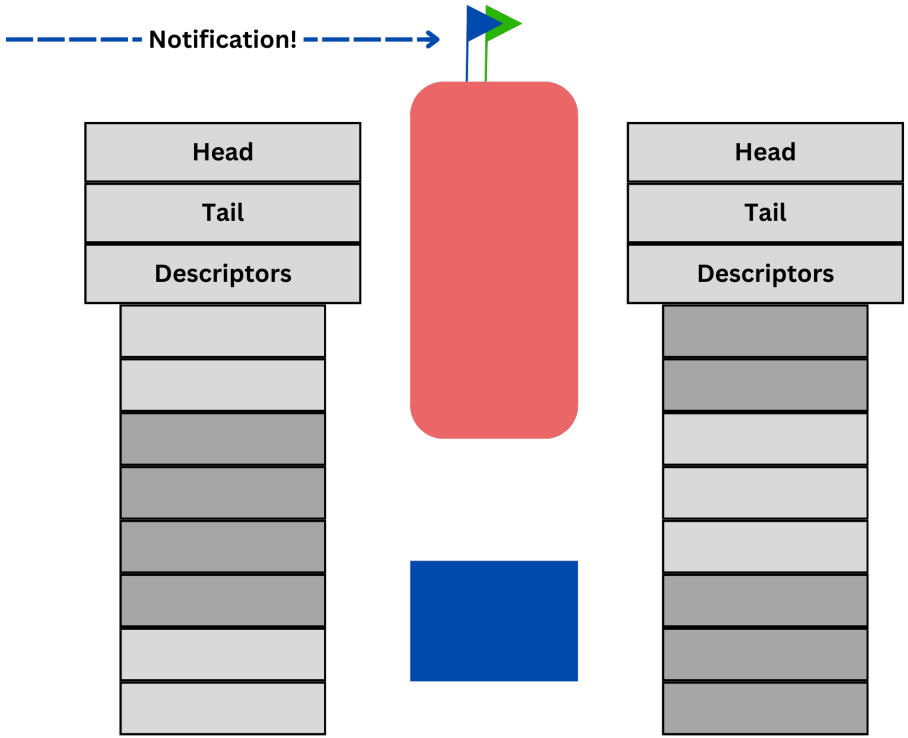
- Asynchronous *notification objects*
- *Shared memory* for data and meta-data transfer
- Subsystem specific *queues*



sDDF Queues

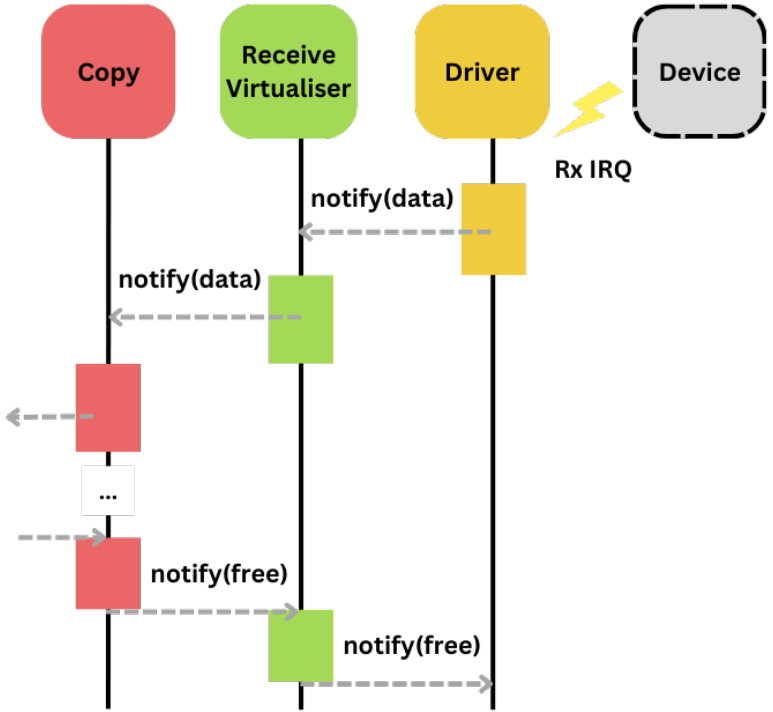
```c
/* buffer descriptor */
typedef struct buff_desc {
    /* offset of buffer within memory region */
    uint64_t offset;
    /* length of data inside buffer */
    uint16_t len;
} net_buff_desc_t;

/* queue */
typedef struct queue {
    /* index to insert at */
    uint16_t tail;
    /* index to remove from */
    uint16_t head;
    /* buffer descriptor array */
    net_buff_desc_t buffers[];
} queue_t;
```

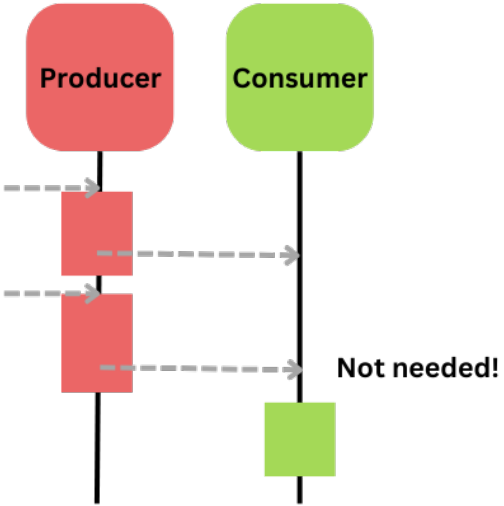


Event Based Components

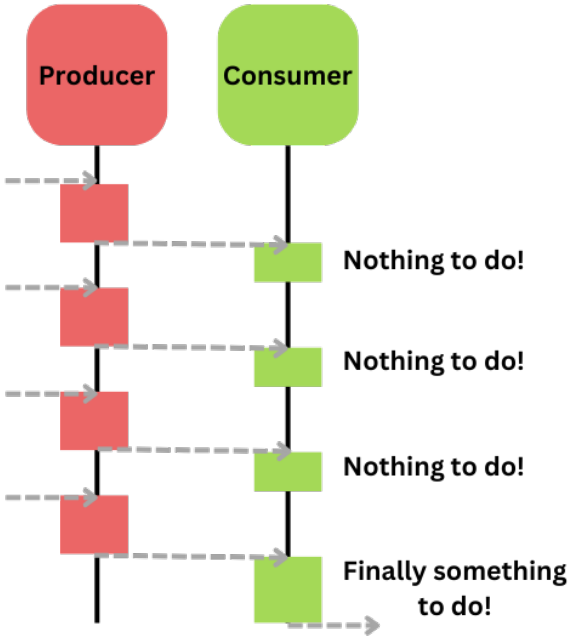
- Components are *blocked* by default
- *Signalling* is required for processing to begin
- This process is called the *signalling protocol*



Signalling Protocols - Over Signalling

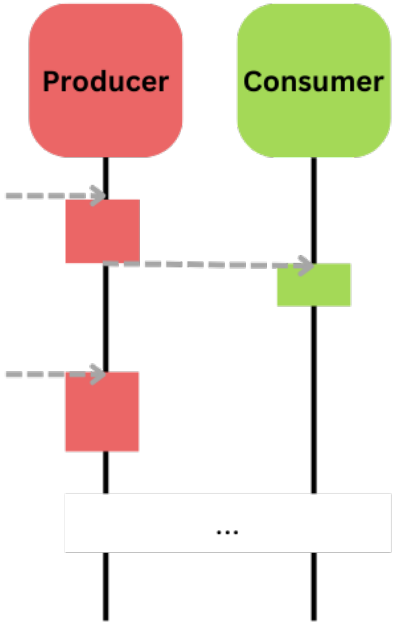


Signalling before a consumer has been scheduled



Signalling when no work can be done

Signalling Protocols - Under Signalling

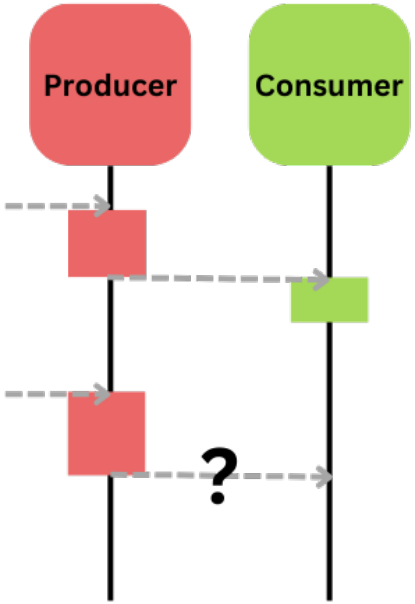


Missed signal - system deadlocked!

Signalling Protocol Development

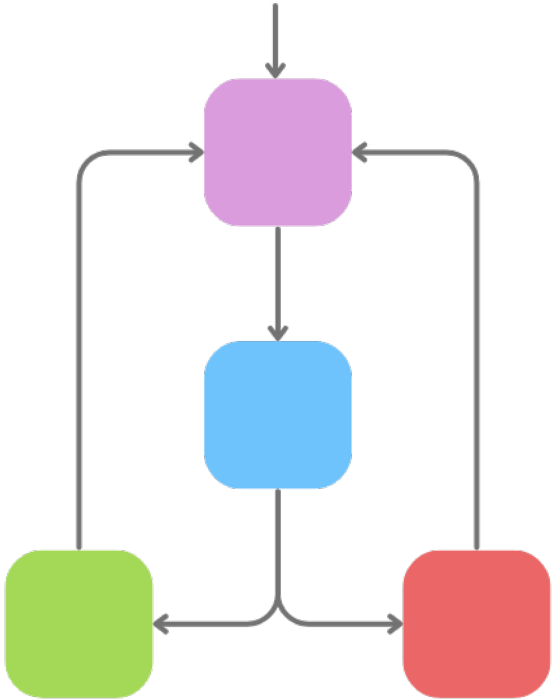
- Components simple but *interleavings* complex
- Reasoning about system state difficult!

➡ *We need more assurance!*



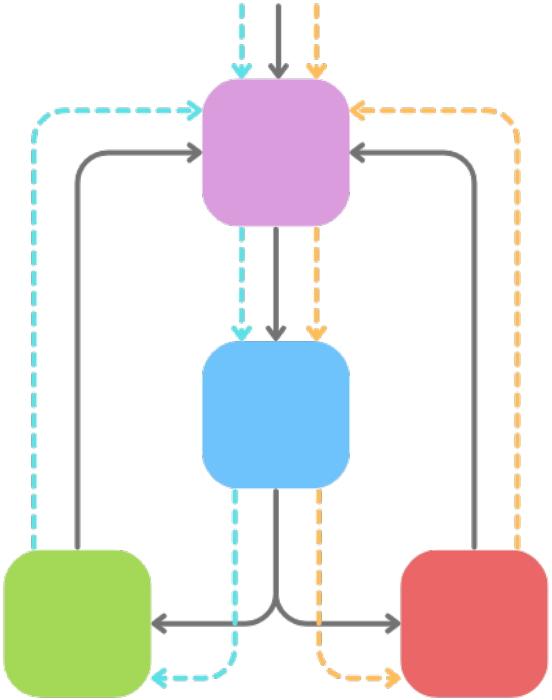
Model Checking - Basics

- Create a *model* of your program
- Model converted into a *directed graph*
- *Paths* of the graph are enumerated
- Each path is examined



Model Checking - Limitations

- *Accuracy* of model must be checked by hand
- Extremely sensitive to *state space explosions*



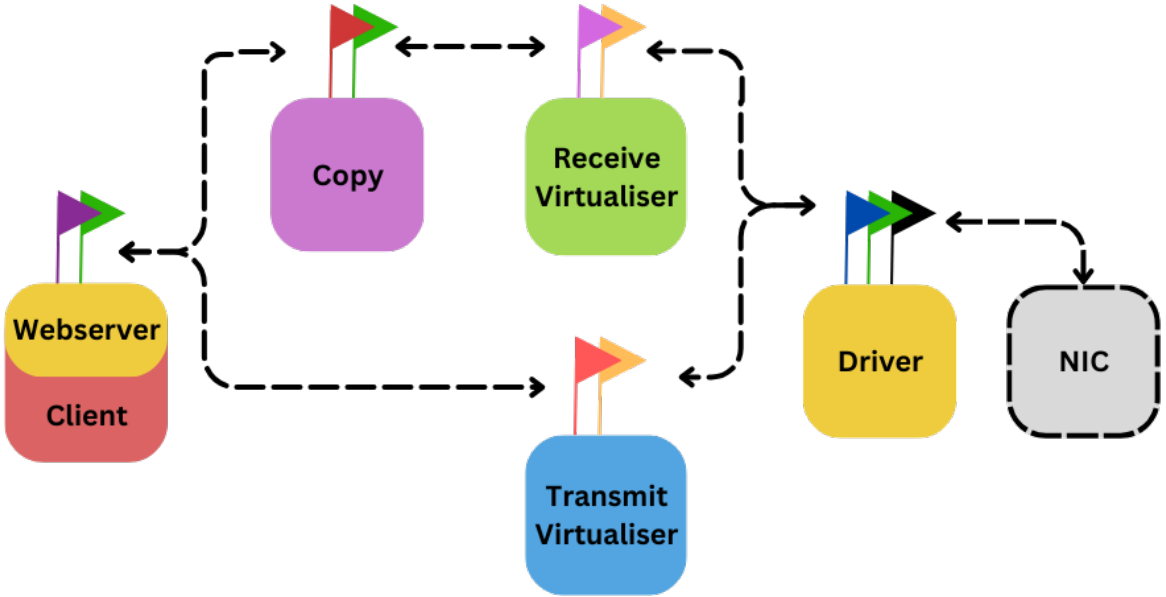
Spin and PROMELA

- The model checker *Spin* was selected
- The modelling language of spin is *PROMELA*

Spin*:* https://spinroot.com/spin/whatispin.html

sDDF Networking

- First device class
- Sensitive to *latency*
- High *throughput* requirements



Modelling Components

- Careful code examination
- Non-essential state abstracted
- Receive and transmit control paths split

  ➡ *Deadlocks found and analysed*

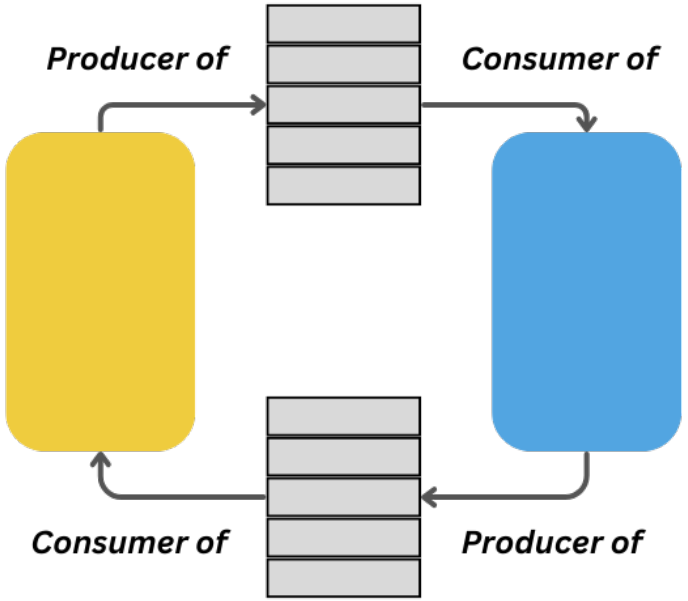
```
// capacity of queue
#define QUEUE_CAPACITY 2

typedef queue {
    // notification object
    chan notification = [1] of {bit};
    // index to remove from
    unsigned head : 2;
    // index to insert at
    unsigned tail : 2;
}
```

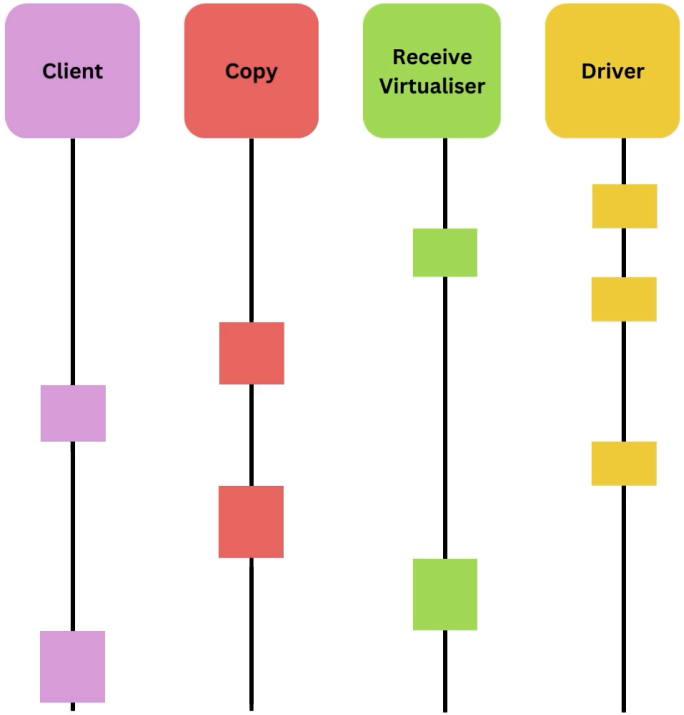
PROMELA queue

Development Begins

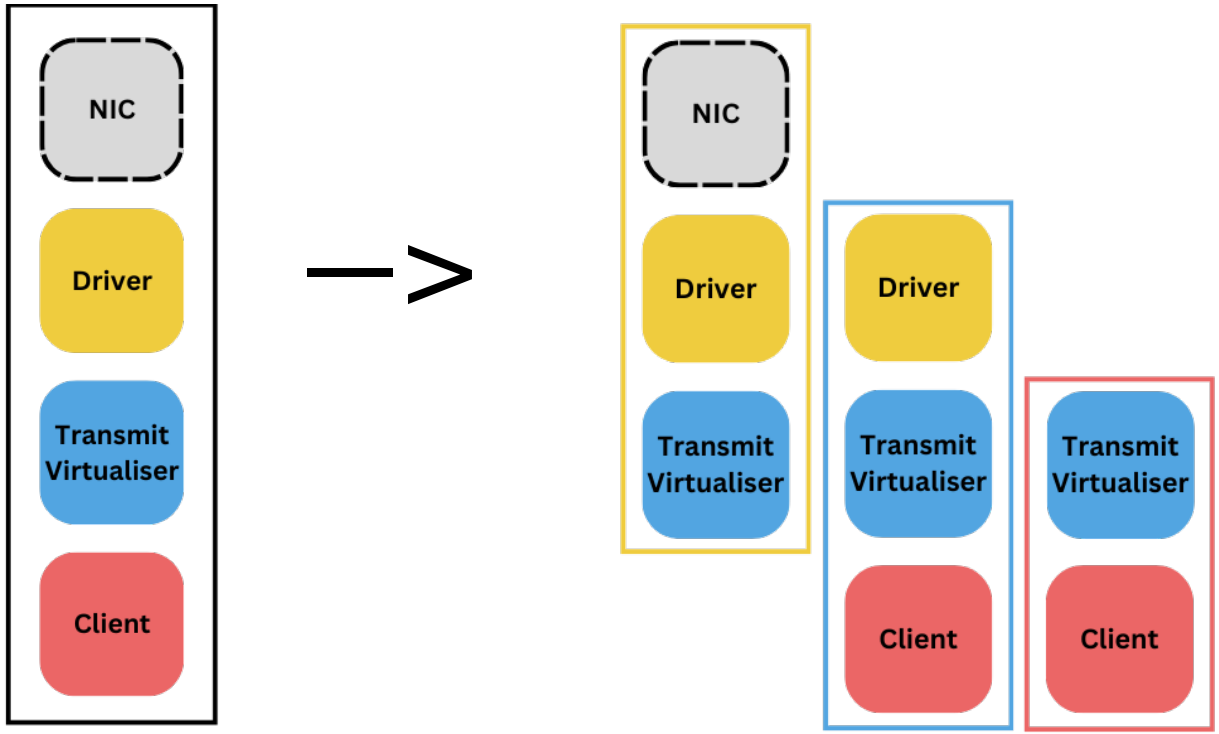
- Experiments using two component system
- Three candidates found and benchmarked
  - ➡ *Most performant protocol selected!*



State-Space Reductions - Priorities

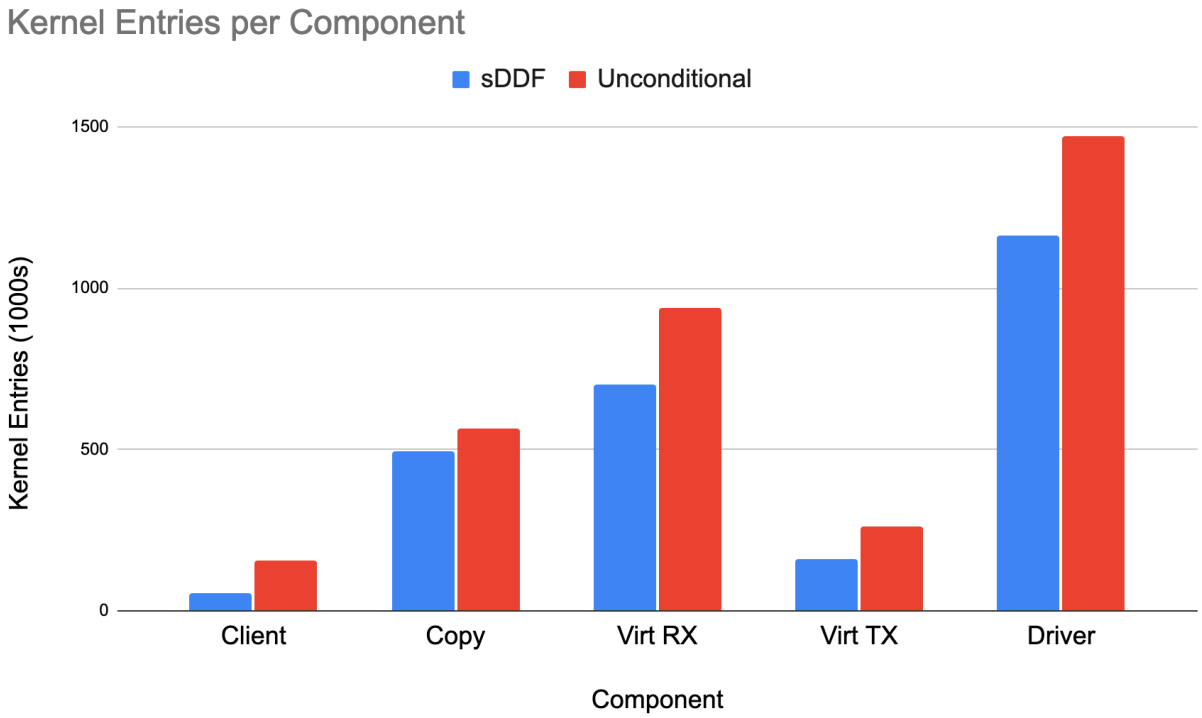


State-Space Reductions - One Component



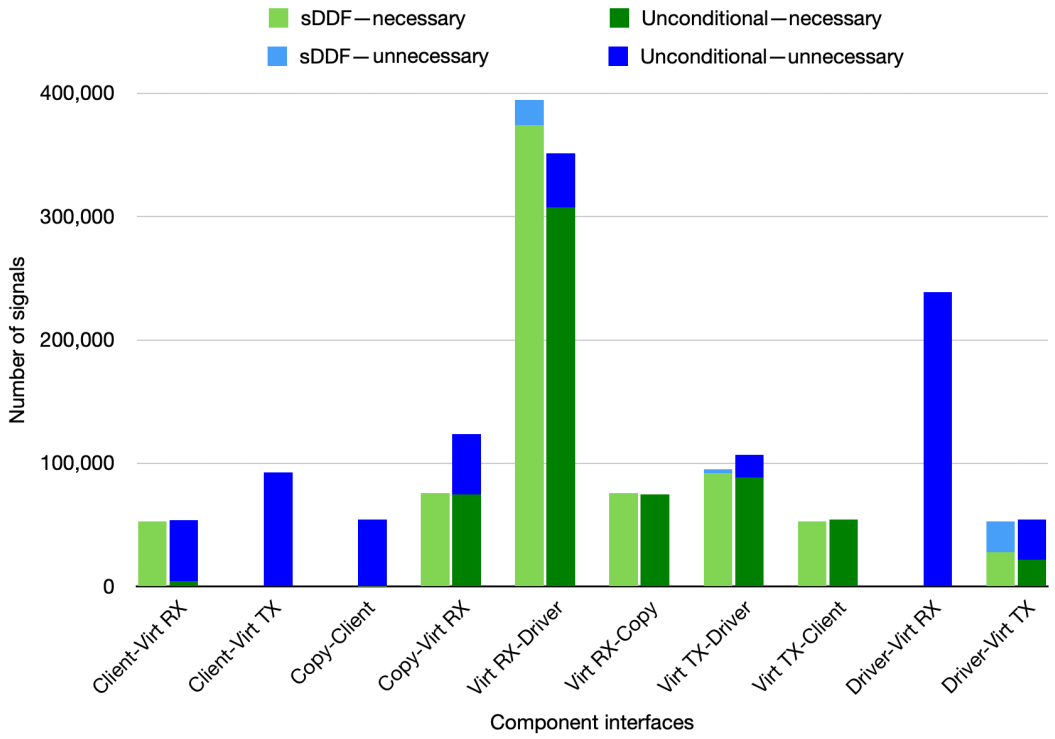
Results - Kernel Entries

- Number of *kernel entries* of each component
- Includes non-signalling kernel entries



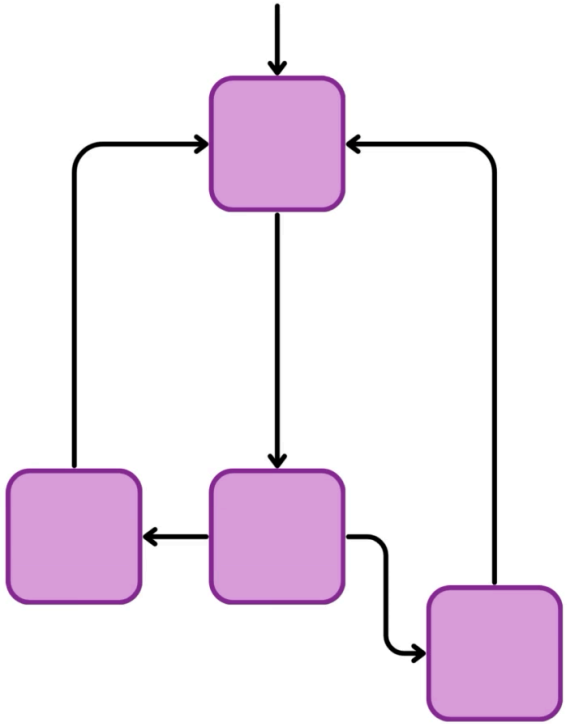
Results - Unnecessary Signals

- Number of times each component is *signalled*
- Number of times there was *no work* from signaller



Future Work

- Prove that the PROMELA model is an *abstraction* of sDDF C code
- Model other subsystems
- Model checking more properties



Links to our Work

PROMELA Modelling: https://github.com/au-ts/sddf/tree/spin_models

Component Models: https://github.com/au-ts/sddf/tree/spin_models/examples/echo_server/spin_models/verified_models

Thank you!