Федеральное государственное бюджетное учреждение науки
Институт системного программирования РАН

Построение EFSM-моделей на основе статического анализа HDL-описаний

Смолов Сергей Александрович

Научный руководитель
д.ф.-м.н. Петренко Александр Константинович

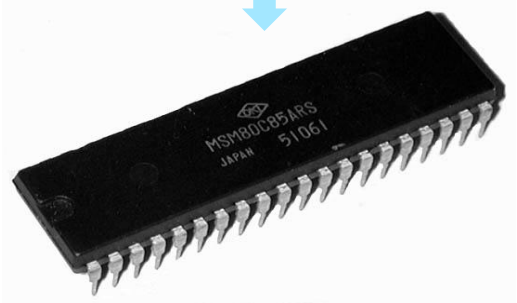
Общая схема создания цифровой аппаратуры

требования

проектирование

HDL-описание

синтез



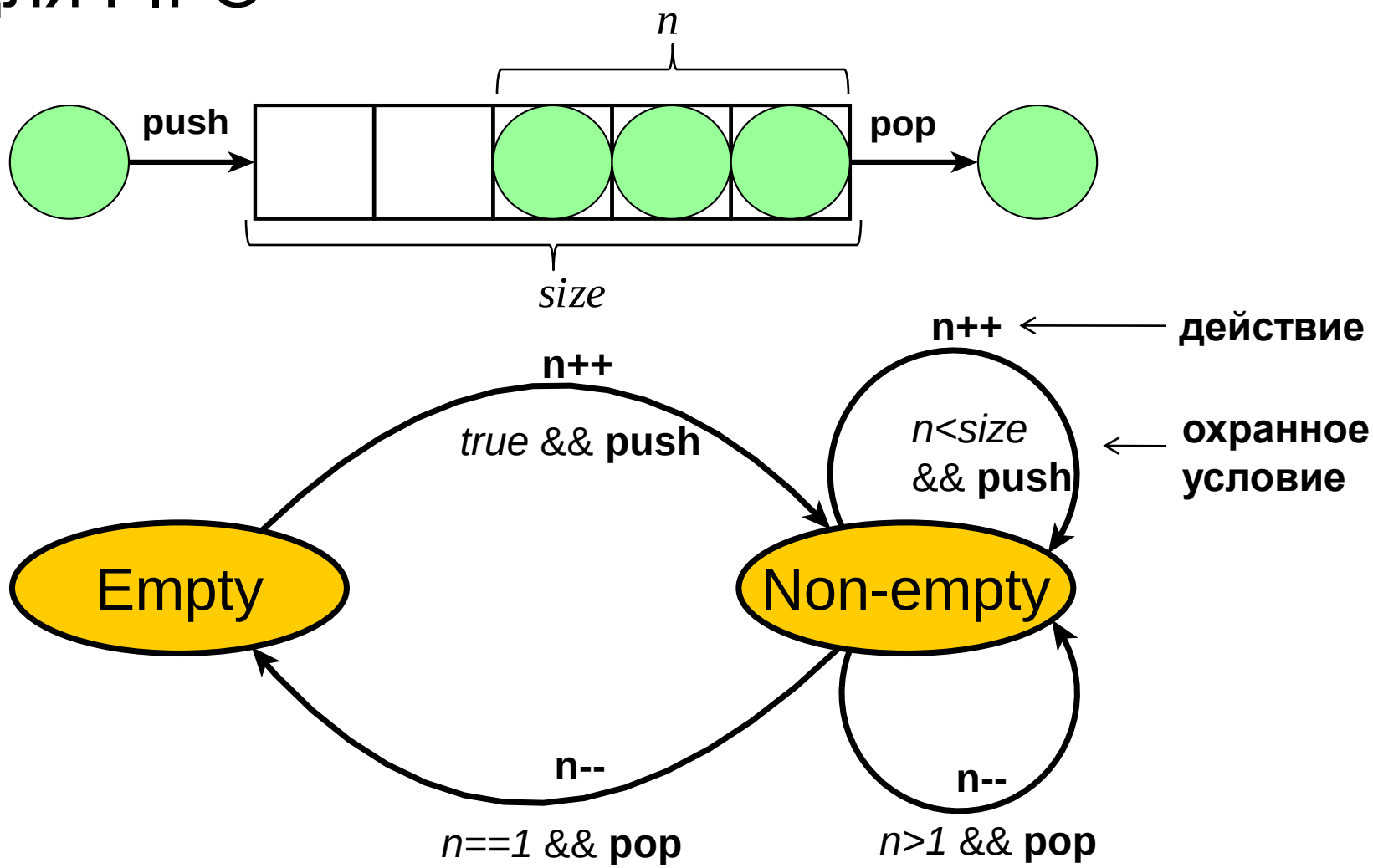
HDL (Hardware Description Language)

- Сходство с распространенными ЯП
- Средства описания структуры (*модули*) и поведения (*процессы*) аппаратуры с потактовой точностью
- Примеры: **VHDL, Verilog**

Некоторые аспекты верификации аппаратуры

- “Узкое место” в процессе проектирования аппаратуры
- Основные методы
  - Формальный
  - Имитационный
- Использование моделей
  - Автоматные модели (FSM, **EFSM**)

Расширенный конечный автомат (EFSM) для FIFO



EFSM-модели в верификации аппаратуры

- Подходят для описания аппаратных систем
  - управляющая логика отделена от функций преобразования данных
- Применимы для
  - формальной проверки свойств
  - генерации тестов, проверяющих соответствие спецификациям

Методы извлечения EFSM-моделей из исходного кода HDL-описаний

- Не пригодны для верификации
  - Логический синтез [Giomi, 1995]
  - Ускорение мутационного тестирования [Bombieri и др., 2012]
- “Веронский метод”
  - Генерация функциональных тестов [Guglielmo и др., 2011]

Основные этапы предлагаемого метода

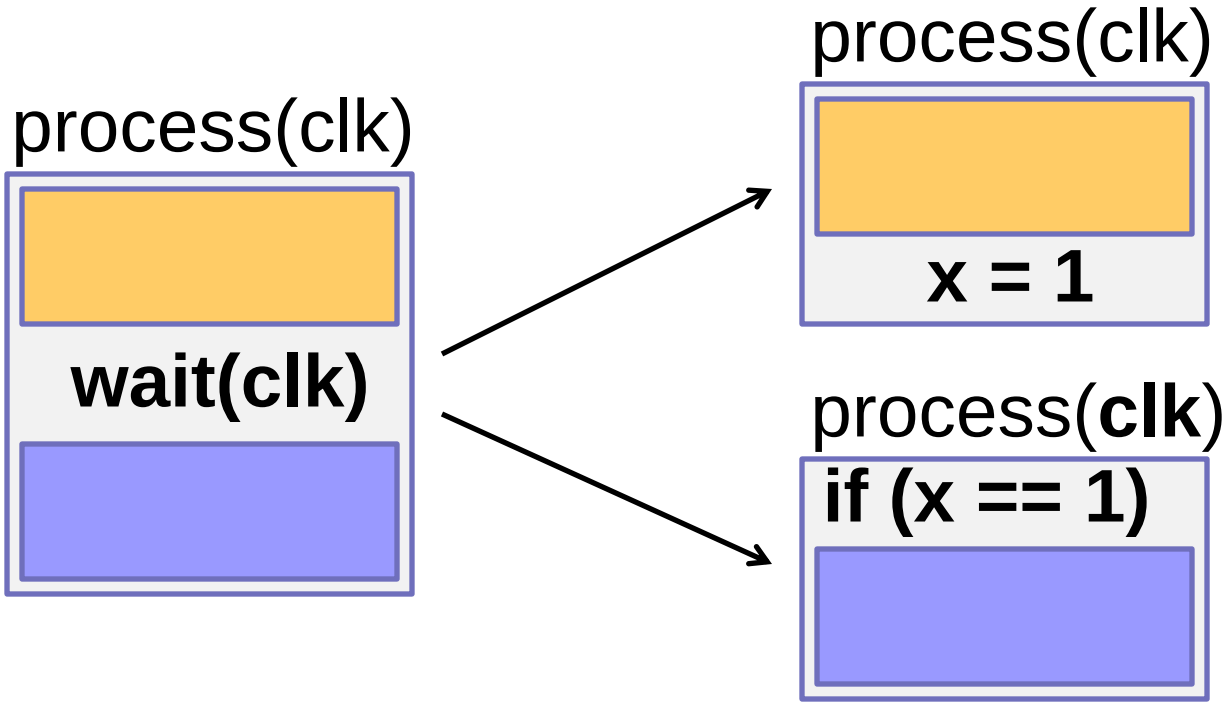
- Предварительная обработка
  - Внутреннее представление
  - Синхросигналы
  - Неявные переменные состояния
- Построение системы охраняемых действий
- Построение состояний и переходов EFSM-модели
  - Явные переменные состояния

Синхросигнал

- Входной 1-битный сигнал
- Присутствует в списке чувствительности (*sensitivity list*) или *wait*-выражении хотя бы одного из процессов
- Не используется в присваиваниях

Неявные переменные состояния

- Декомпозиция многотактных процессов на однотактные микрооперации



Извлечение охраняемых действий

- Охраняемое действие – пара $\gamma \rightarrow \delta$
  - γ - охранное условие, δ - действие
- Для каждого процесса p извлекается множество $\{<C_p^{(i)}, \gamma_p^{(i)} \rightarrow \delta_p^{(i)}>\}_{i=1,n}$
  - $C_p^{(i)}$ – множество синхросигналов
  - $\gamma_p^{(i)}$ – условие верхнего уровня процесса p (если нет, то, $\gamma_p^{(i)} \equiv true$)
  - $\delta_p^{(i)}$ – инструкции p

Извлечение охраняемых действий на примере счетчика

```
process (…) begin
  if clock'event then
    if reset = '1' then
      state <= '0';
    elsif en = '1' then
      state <= state +'1';
    end if;
  end if;
  count <= state;
end process;
```

C1	{*clock*}
γ1	(*reset* = 1)
δ1	*state* <= 0
C2	{*clock*}
γ2	!(reset=1)&&(en=1)
δ2	state <= state + 1
C3	∅
γ3	true
δ3	count <= state

Анализ охраняемых действий

- v – переменная, x, y – охраняемые действия ($\gamma_x \rightarrow \delta_x$, $\gamma_y \rightarrow \delta_y$)
- v *определяется* в x ($v \in Def_x$), если δ_x содержит присваивание v
- v *используется* в y ($v \in Use_y$), если v присутствует в γ_y или в правой части некоторого присваивания δ_y
- y *зависит* от x, если $Def_x \cap Use_y \neq \varnothing$
  - *зависимости по управлению*
  - *зависимости по данным*

x - переменная состояния (state), если:

- Не является входным сигналом
- Существует охраняемое действие, которое через *x* зависит по управлению от самого себя
- Все охраняемые действия, которые используют *x*, синхронизируются одинаково

Построение состояний EFSM-модели

- Построение графа зависимостей между охраняемыми действиями
- Определение множества state и его факторизация
- Извлечение ограничений на state из охранных условий
- Построение множества попарно несовместных **состояний**

Построение переходов EFSM-модели

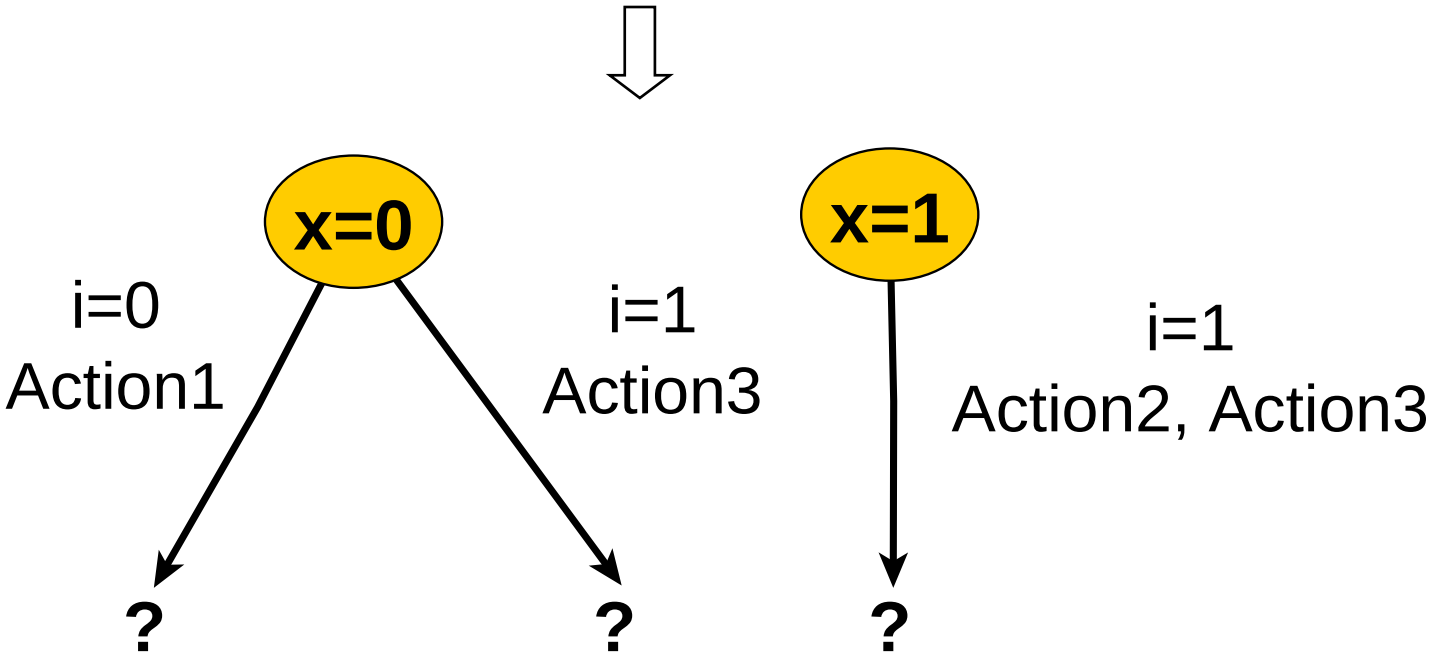
- Начальные состояния
  - Анализ совместности состояний и охранных условий
- Конечные состояния
  - Совпадение действий и состояний
  - “Расщепление” переходов + уточнение охранных условий

Построение EFSM-модели (1)

γ_1: x==0 && i==0
δ_1: Action1

γ_2: x==1 && i==1
δ_2: Action2

γ_3: (x==0 || x==1) && i== 1
δ_3: Action3



Построение EFSM-модели (2)

- Action1: $x = i$

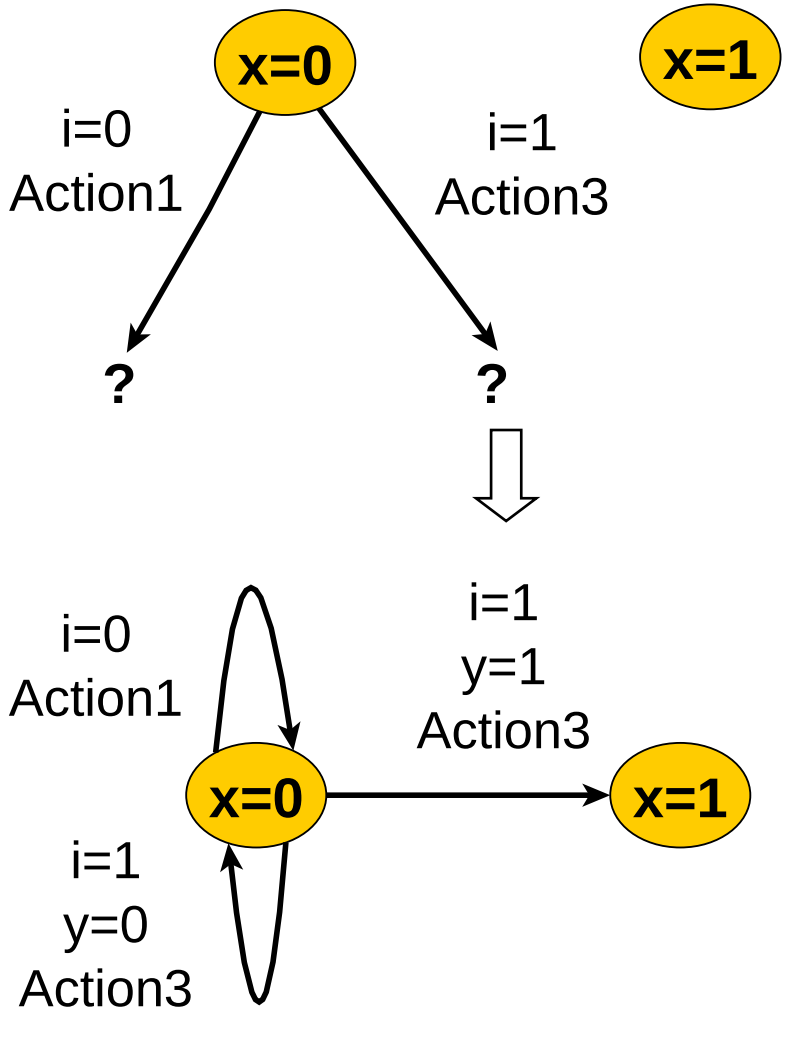
Конец перехода:

$\{i = 0 \;\&\&\; x = i\} \rightarrow \{x = 0\}$

- Action3: $x = \sim y$

Охранные условия:

$\{y = 0\}, \{y = 1\}$



Промежуточные результаты

- Предложен новый метод извлечения EFSM-моделей
- Разработано внутреннее представление для охраняемых действий и EFSM-моделей
- Эвристики определения синхросигналов и переменных состояния апробированы на ряде модулей на языке VHDL

Направления дальнейших исследований

- Апробация
- Имитационная верификация
  - Построение шаблонов тестового окружения
- Формальная верификация
  - Поиск состояний гонки, зависаний, тупиков
  - Полное покрытие кода

Спасибо!