FPL 2010
milano, italy

Optimization of Regular Expression Pattern Matching Circuit Using At-Most Two-Hot Encoding on FPGA

SangKyun Yun* and KyuHee Lee

Dept. of Computer & Telecomm. Eng.
Yonsei University, Wonju, Korea

YONSEI UNIVERSITY

In this paper

- propose a new state encoding scheme, called **At-Most Two-Hot (AMTH)** encoding
- FPGAs such as Virtex-5 and 6 offer **six-input LUT**s (**6-LUT**s). AMTH encoding increases the utilization of inputs of 6-LUT.
- optimize regular expression pattern matching circuit using AMTH encoding on **FPGA with 6-LUT**s

Introduction

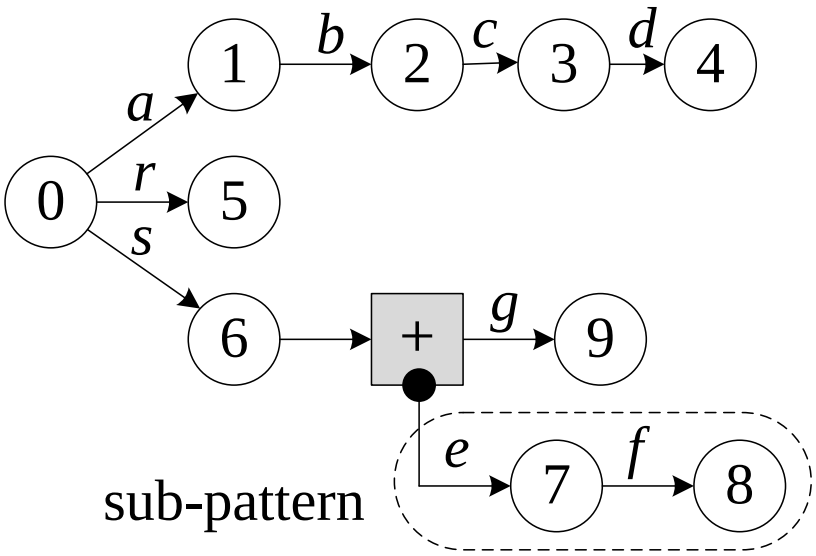
- Regular Expression
  - widely used to represent attack patterns in network intrusion detection systems(NIDS)

- Hardware based regular expression matching
  - **FPGA based implementation** – NFA
  - Memory based implementation – DFA

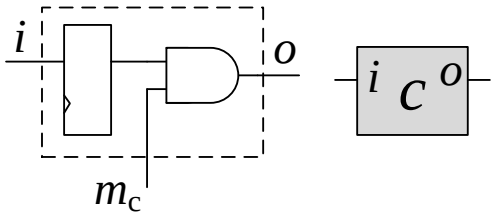
Basic Implementation of NFA-based matching

- Example Patterns: ***abcd,*** ***r,*** ***s(ef)+g***

- NFA (Pattern Tree)

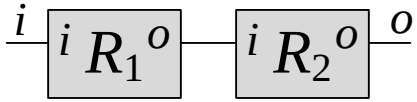


Basic Building Blocks

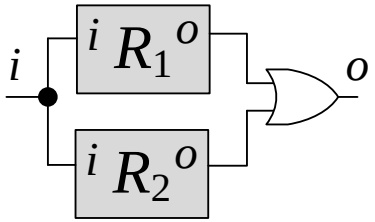


one-hot encoding

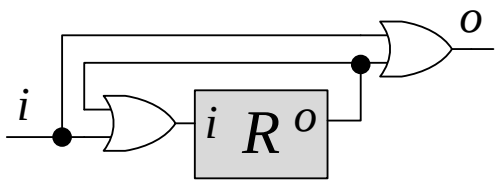
(a) c



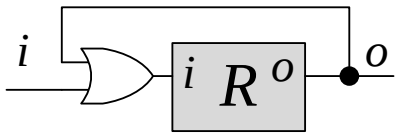
(b) $R_1 \bullet R_2$



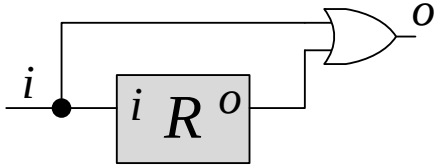
(c) $R_1 \mid R_2$



(d) R^*



(e) $R+$

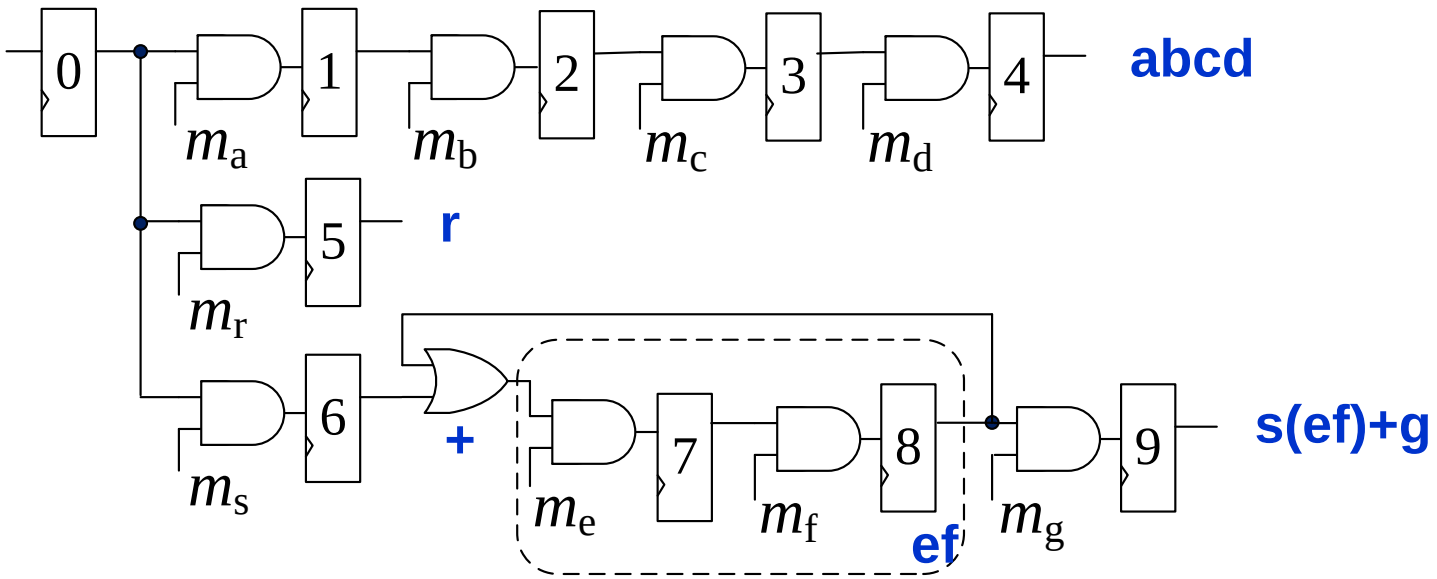


(f) $R?$

m_c : output of shared character decoder for input character **c**
(1 when an input character is **c**)

R, R_1, R_2 : regular expressions

One-hot Encoded Implementation



10 flip-flops

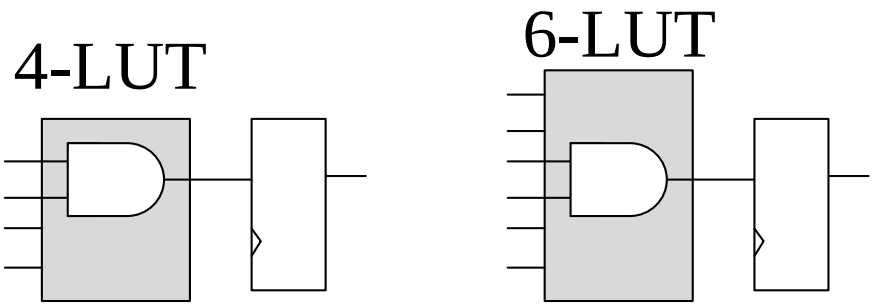
Optimization Methods

- common prefix sharing [Hutchings'02]
- shared character decoder [Clark'03]
- common infix sharing [Lin'07]
- building blocks for constraint repetitions [Bispo'06]

➔ Their implementations adopted **one-hot** encoding scheme for state assignment

Motivation

- Increasing the number of inputs
  - conventional FPGAs provide **4-LUT**s
  - recently announced FPGAs such as Vertex-5 and Vertex-6 provides **6-LUT**s



- **4-LUT**s are sufficient for **one-hot** encoded implementation of regular expression matching circuits
  ➔ additional inputs of **6-LUT**s may be **wasteful**

State encoding schemes

- One-hot encoding
  - ***N*** states ➔ use ***N*** flip-flops
  - suited to register-rich FPGA architecture
  - however, additional inputs of 6-LUT may be wasteful.

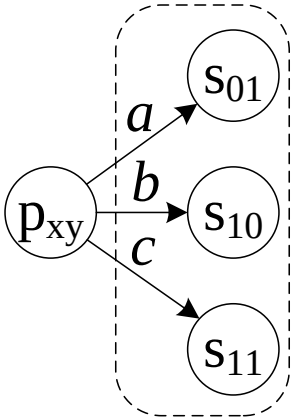
- Binary encoding
  - ***N*** states ➔ use $\log_2 N$ flip-flops
  - requires multi-level LUT logic
    → inefficient in FPGA implementation

- Need a new state encoding scheme
  - to increase utilization of inputs of 6-LUTs
  - without the degradation of performance

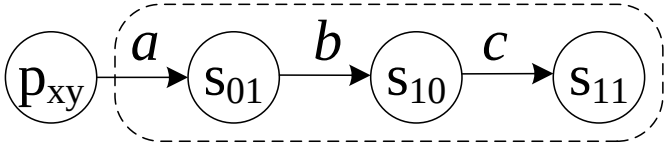
At-Most Two-Hot (AMTH) Encoding

- Basic Idea of AMTH encoding
  - two flip-flops are associated with three states.
  - one or two flip-flops can have value 1 for each state
    - 01, 10, 11 → three states
    - 00 → all of three states are inactive
- In the state machine of three states, if the combinational logic can be implemented in two 6-LUTs, the three states can be implemented using two logic elements (LE)

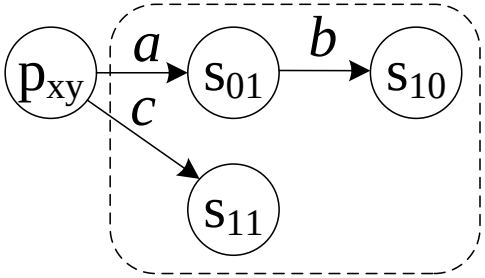
Four types of AMTH encoding groups



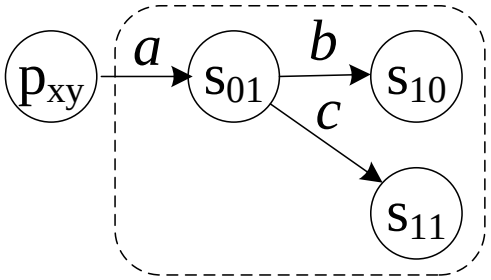
(a) type 1



(b) type 4

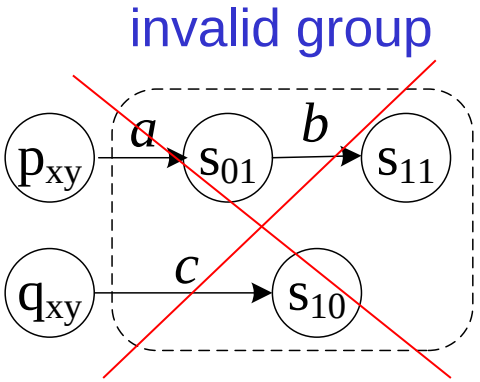


(c) type 2



(d) type 3

invalid group



- each encoding group has only one previous state.

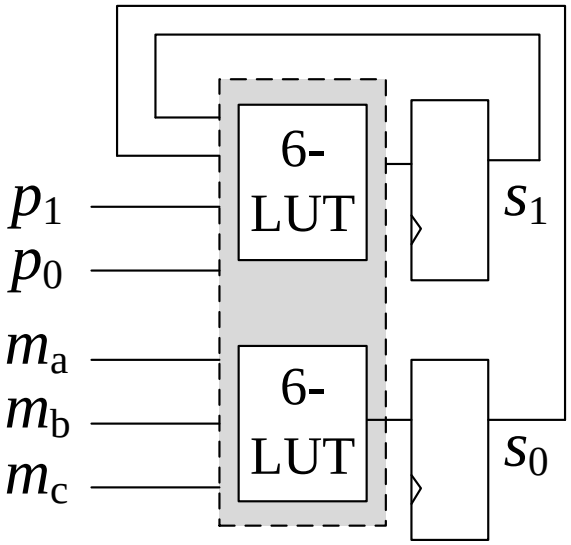
State Transition Equation of each type

type	Equations	# inputs
1	$S_1 <= (p_1p_0=xy) \cdot m_b + (p_1p_0=xy) \cdot m_c$ $S_0 <= (p_1p_0=xy) \cdot m_a + (p_1p_0=xy) \cdot m_c$	4 4
2	$S_1 <= (s_1s_0=xy) \cdot m_b + (p_1p_0=xy) \cdot m_c$ $S_0 <= (p_1p_0=xy) \cdot m_a + (p_1p_0=xy) \cdot m_c$	**6** 4
3	$S_1 <= (s_1s_0=xy) \cdot m_b + (s_1s_0=xy) \cdot m_c$ $S_0 <= (p_1p_0=xy) \cdot m_a + (s_1s_0=xy) \cdot m_c$	4 **6**
4	$S_1 <= (s_1s_0=xy) \cdot m_b + (s_1s_0=xy) \cdot m_c$ $S_0 <= (p_1p_0=xy) \cdot m_a + (s_1s_0=xy) \cdot m_c$	4 **6**

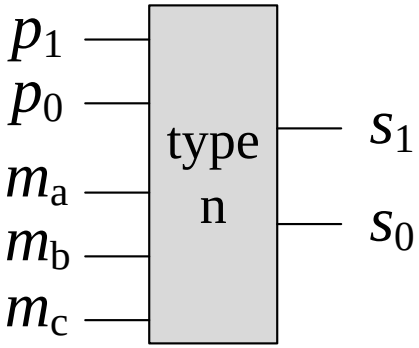
→ implemented in two 6-LUTs

YONSEI UNIVERSITY

AMTH implementation using 6-LUTs

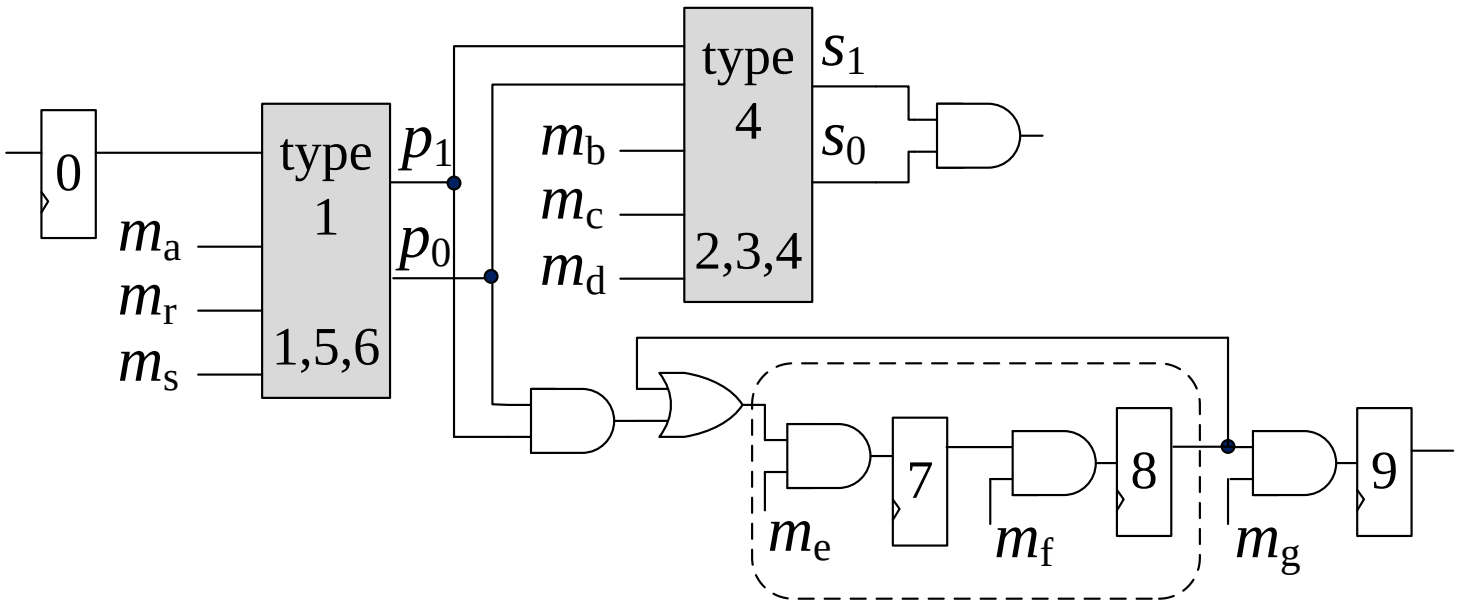


(a) implementation



(b) block diagram

AMTH encoded Implementation of **p.6** circuit



10 states → 8 flip-flops

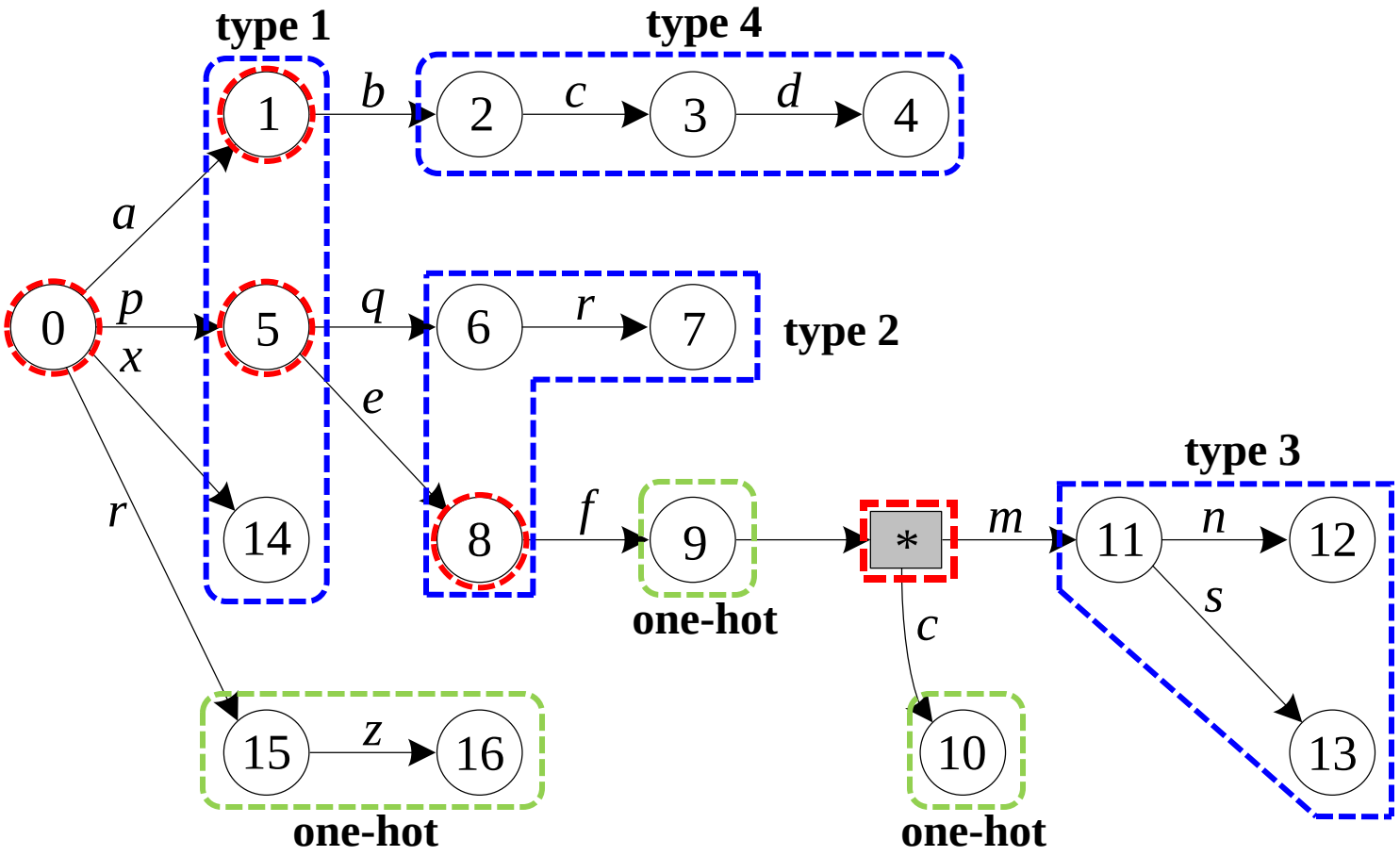
YONSEI UNIVERSITY

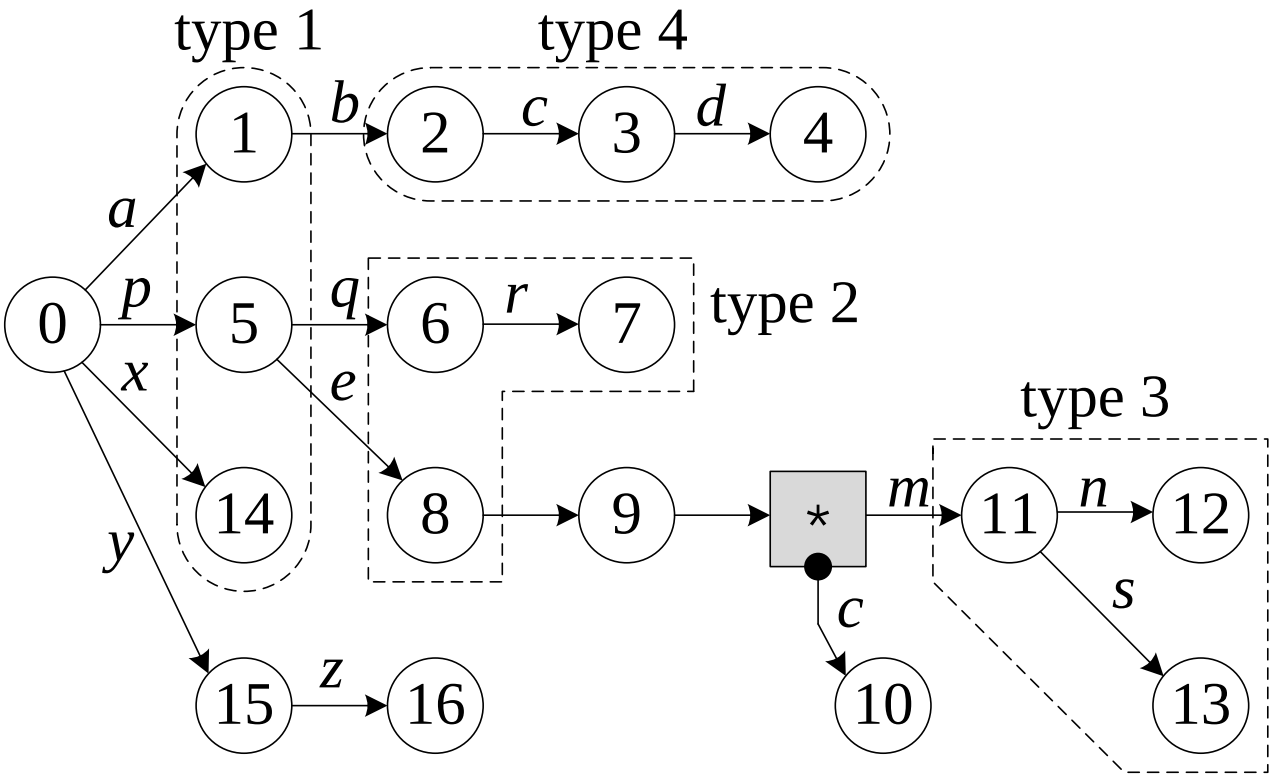
AMTH encoding procedure of pattern tree

- AMTH encoding procedure
  1. A **root** state of a pattern tree is implemented using **one-hot** encoding
  2. Let root be a current node
  3. At first, find **type 1 groups** at current node and then encode them in AMTH
  4. Find the **type 2, type3, and type 4 groups** and then encode them in AMTH
  5. The remaining **one or two states** are encoding using **one-hot** encoding
  6. For each of newly encoded states, let the state be a current node and repeat from step 3 to step 5
- meta-characters split a pattern into sub-patterns
  - join them after sub-patterns are encoded

Example: AMTH Encoding Procedure

- patterns : ***abcd, pqr, pefc*mn, pefc*ms, x, yz***



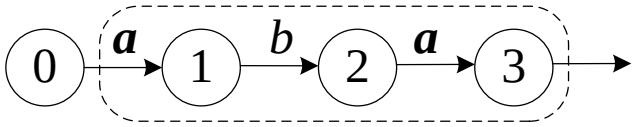


17 states → 5 + 4*2 = 13 flip-flops

YONSEI UNIVERSITY

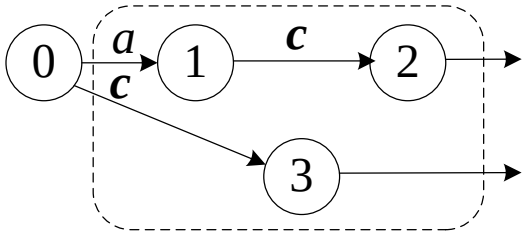
Overlapped Matching

- overlapped matching in NFA for regular exp. matching
  - several states (except initial state) can be simultaneously active since initial state is always active during pattern matching.
- examples



-	1	0	0	0
a	1	1	0	0
b	1	0	1	0
a	1	**1**	0	**1**

input sequence: aba ...



-	1	0	0	0
a	1	1	0	0
c	1	0	**1**	**1**

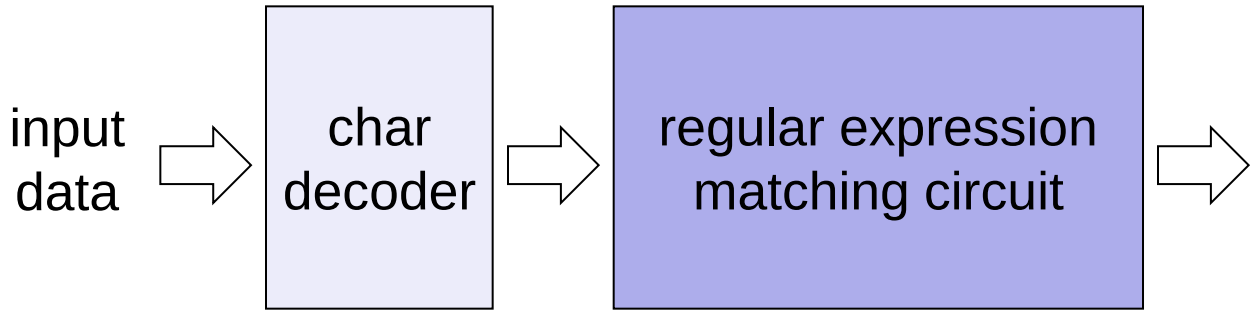
input sequence: ac ...

Handling Overlapped Matching

- In AMTH encoding, overlapped matching states should not belong to the same group.
- Overlapped matching is possible in type 2, 3, 4 group

- Modification of AMTH encoding procedure
  - Find non-overlapped matching groups
  - If only overlapped matching groups remain, one hot encoding is used for overlapped matching groups

Evaluation



- Snort rule v2.8 (Sep. 2008)
  - 1,845 pure regular expression patterns (PCRE)
  - 3,267 static patterns (content)
- The regular expression matching circuits described in Verilog are generated by automatic circuit generation program
- synthesized on Xilinx Virtex-5 FPGA with 500MHz max clock freq. using ISE 11.1

Result of FPGA Synthesis

- pure regular expression patterns

encoding	# rules	# states	# FFs	# CLBs	f_{max}(MHz)
One-Hot	1,845	52,551	52,551	7,290	402.36
AMTH			**40,460**	5,884	402.36
			77% (23% ↓)	81% (19% ↓)	

- static patterns

encoding	# rules	# states	# FFs	# CLBs	f_{max}(MHz)
One-Hot	3,267	28,003	28,003	4,073	993.20*
AMTH			**19,599**	3,199	792.06*
			70% (30% ↓)	78.5% (21.5% ↓)	* actual f_{max} = 500.0 MHz

YONSEI UNIVERSITY

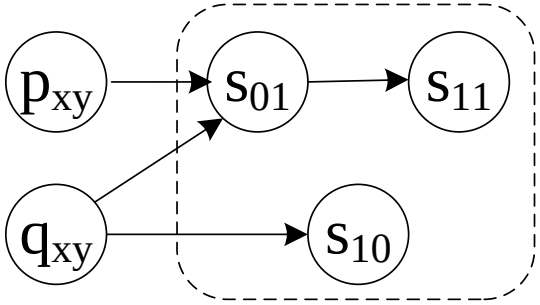
Conclusion

- propose At-Most Two-Hot(AMTH) state encoding scheme to increase the utilization of 6-LUTs
- AMTH encoding can be used in the optimization of regular expression pattern matching circuit on FPGA with 6-LUTs
- AMTH encoding can ideally reduce the required logic elements up to 33% (when no one-hot encoding)
- In the implementation of regular expression pattern matching circuit, AMTH encoding provides 23-30% LE savings, 19-21% CLB savings, comparing to one-hot encoding

YONSEI UNIVERSITY

Further Study

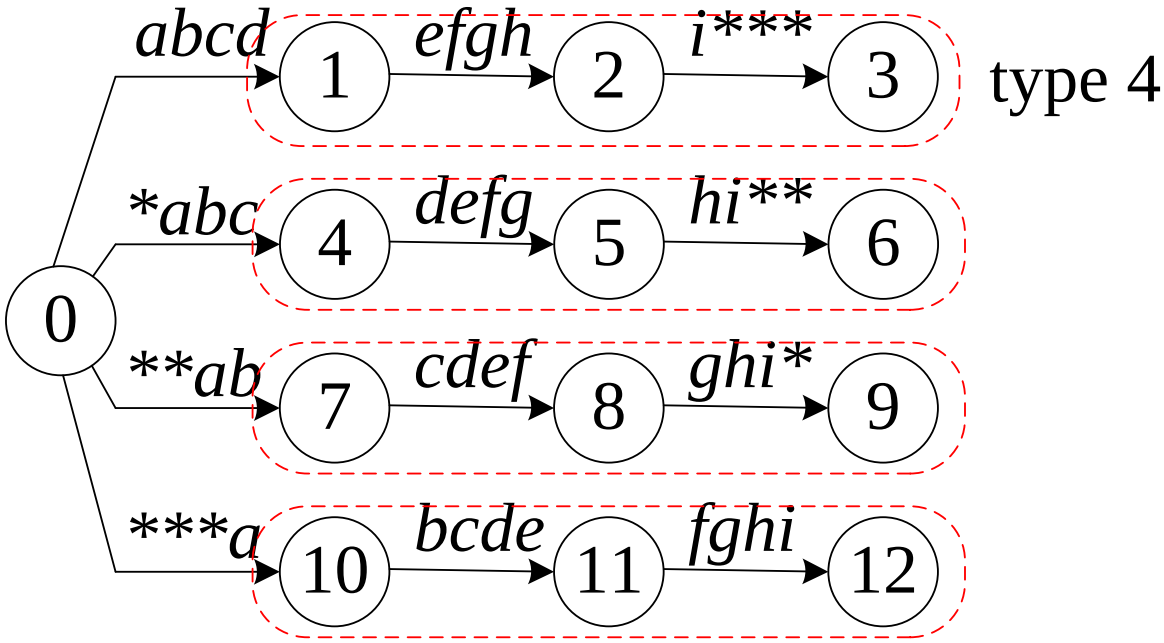
- AMTH encoding scheme can be applied to the implementation of general FSM.
  - each group may have more than one previous states.
    → may not be implemented in one-level 6-LUT logic.



- AMTH encoding can be also used in multi-byte based regular expression pattern matching
- Generalization of AMTH
  → At most **k**-hot encoding can be also considered

Multi-byte processing using AMTH encoding

- pattern: ***abcdefghi***



4-byte processing at a time

Generalized At-Most k-Hot encoding

- k flip-flops represent up to 2^k-1 states
  - all zeroes means that all of 2^k-1 states are inactive.
- state transition equations require at most 2k + 2^{k-1} inputs (k>=2)
  - k bit parent state
  - k bit current state
  - at most 2^{k-1} inputs
- the number of inputs

  k=2: 4+2 = 6 → **one level 6-LUT**

  k=3: 6+4 = 10 → two level 6-LUT

  k=4: 8+8 = 16 ... → three level 6-LUT