

# State Machine Encoding

Last updated 1/22/21

# State Machine Encoding

- These slides introduce the various methods for encoding states
- Upon completion: You should be able to encode states in any of the described formats

# State Machine Encoding

- Encoding

**Definition of ENCODE (WEBSTER)**

transitive verb

**1a** : to convert (as a body of information) from one system of communication into another; *especially* : to convert (a message) into code

- Morse Code
- Braille

# State Machine Encoding

- Encoding
  - Binary (standard)
  - Represent a set of values (states, light colors, directions ...) with a standard binary sequence
    - $S_0 \rightarrow 00$ ,  $S_1 \rightarrow 01$ ,  $S_2 \rightarrow 10$ ,  $S_3 \rightarrow 11$
    - Red  $\rightarrow 00$ , Yellow  $\rightarrow 01$ , Green  $\rightarrow 10$
    - Forward  $\rightarrow 00$ , Turn right  $\rightarrow 01$ , Turn left  $\rightarrow 10$
  - $m$  values require  $\text{ceil}(\log_2 m)$  code bits
  - $n$  code bits can represent  $2^n$  values

# State Machine Encoding

- Encoding
  - Binary
  - Stop light example

Current State S	Inputs		Next State S'
	T <sub>NS</sub>	T <sub>EW</sub>	
ST0	0	x	ST1
ST0	1	x	ST0
ST1	x	x	ST2
ST2	x	0	ST3
ST2	x	1	ST2
ST3	x	x	ST0

Current State S	Current State		Inputs		Next State	
	S1	S0	T <sub>NS</sub>	T <sub>EW</sub>	S1'	S0'
ST0	0	0	0	x	0	1
ST0	0	0	1	x	0	0
ST1	0	1	x	x	1	0
ST2	1	0	x	0	1	1
ST2	1	0	x	1	1	0
ST3	1	1	x	x	0	0





# State Machine Encoding

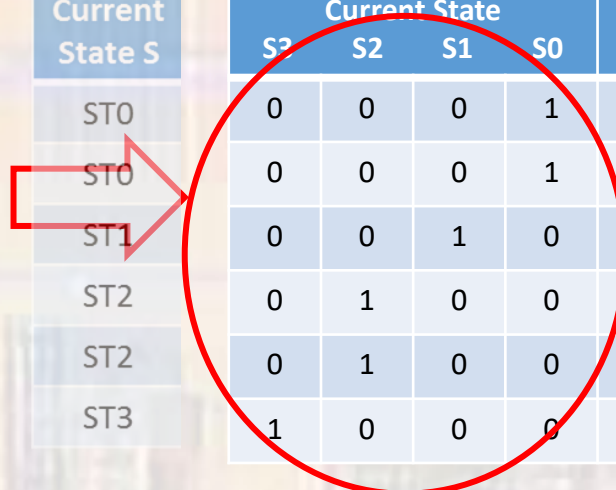
- Encoding
  - One Hot
  - Represent a set of values (states, light colors, directions ...) with a binary word with only one bit high for each code
    - $S_0 \rightarrow 0001$ ,  $S_1 \rightarrow 0010$ ,  $S_2 \rightarrow 0100$ ,  $S_3 \rightarrow 1000$
    - Red  $\rightarrow 001$ , Yellow  $\rightarrow 010$ , Green  $\rightarrow 100$
    - Forward  $\rightarrow 001$ , Turn right  $\rightarrow 010$ , Turn left  $\rightarrow 100$
  - m values require m code bits
  - n code bits can represent n values

# State Machine Encoding

- Encoding
  - One-hot
  - Stop light example

Current State S	Inputs		Next State S'
	T <sub>NS</sub>	T <sub>EW</sub>	
ST0	0	x	ST1
ST0	1	x	ST0
ST1	x	x	ST2
ST2	x	0	ST3
ST2	x	1	ST2
ST3	x	x	ST0

Current State S	Current State				Inputs		Next State			
	S3	S2	S1	S0	T <sub>NS</sub>	T <sub>EW</sub>	S3'	S2'	S1'	S0'
ST0	0	0	0	1	0	x	0	0	1	0
ST0	0	0	0	1	1	x	0	0	0	1
ST1	0	0	1	0	x	x	0	1	0	0
ST2	0	1	0	0	x	0	1	0	0	0
ST2	0	1	0	0	x	1	0	1	0	0
ST3	1	0	0	0	x	x	0	0	0	1



# State Machine Encoding

- Encoding
  - One Not (one cold)
  - Represent a set of values (states, light colors, directions ...) with a binary word with only one bit low for each code
    - $S_0 \rightarrow 1110$ ,  $S_1 \rightarrow 1101$ ,  $S_2 \rightarrow 1011$ ,  $S_3 \rightarrow 0111$
    - Red  $\rightarrow 110$ , Yellow  $\rightarrow 101$ , Green  $\rightarrow 011$
    - Forward  $\rightarrow 110$ , Turn right  $\rightarrow 010$ , Turn left  $\rightarrow 011$
  - m values require m code bits
  - n code bits can represent n values

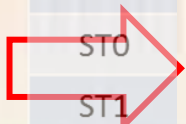


# State Machine Encoding

- Encoding
  - One-hot (one-cold)
  - Stop light example

Current State S	Inputs		Next State S'
	T <sub>NS</sub>	T <sub>EW</sub>	
ST0	0	x	ST1
ST0	1	x	ST0
ST1	x	x	ST2
ST2	x	0	ST3
ST2	x	1	ST2
ST3	x	x	ST0

Current State S	Current State				Inputs		Next State			
	S3	S2	S1	S0	T <sub>NS</sub>	T <sub>EW</sub>	S3'	S2'	S1'	S0'
ST0	1	1	1	0	0	x	1	1	0	1
ST0	1	1	1	0	1	x	1	1	1	0
ST1	1	1	0	1	x	x	1	0	1	1
ST2	1	0	1	1	x	0	0	1	1	1
ST2	1	0	1	1	x	1	1	0	1	1
ST3	0	1	1	1	x	x	1	1	1	0



# State Machine Encoding

- Encoding
  - Gray
  - Represent a set of values (states, light colors, directions ...) with a binary word with only one bit different between adjoining codes - twisted shift register
    - $S_0 \rightarrow 00$ ,  $S_1 \rightarrow 10$ ,  $S_2 \rightarrow 11$ ,  $S_3 \rightarrow 01$
    - $A \rightarrow 000$ ,  $B \rightarrow 001$ ,  $C \rightarrow 011$ ,  $D \rightarrow 010$ ,  
 $E \rightarrow 110$ ,  $F \rightarrow 111$ ,  $G \rightarrow 101$ ,  $H \rightarrow 100$
  - $m$  values require  $\text{ceil}(\log_2 m)$  code bits
  - $n$  code bits can represent  $2^n$  values

# State Machine Encoding

- Encoding
  - Gray
  - Stop light example

Current State S	Inputs		Next State S'
	T <sub>NS</sub>	T <sub>EW</sub>	
ST0	0	x	ST1
ST0	1	x	ST0
ST1	x	x	ST2
ST2	x	0	ST3
ST2	x	1	ST2
ST3	x	x	ST0

Current State S	Current State		Inputs		Next State	
	S1	S0	T <sub>NS</sub>	T <sub>EW</sub>	S1'	S0'
ST0	0	0	0	x	0	1
ST0	0	0	1	x	0	0
ST1	0	1	x	x	1	1
ST2	1	1	x	0	1	0
ST2	1	1	x	1	1	1
ST3	1	0	x	x	0	0



# State Machine Encoding

- Encoding
  - Johnson
  - Represent a set of values (states, light colors, directions ...) with a binary word with only one bit different between adjoining codes - twisted shift register
    - $S_0 \rightarrow 00$ ,  $S_1 \rightarrow 10$ ,  $S_2 \rightarrow 11$ ,  $S_3 \rightarrow 01$
    - $A \rightarrow 000$ ,  $B \rightarrow 100$ ,  $C \rightarrow 110$ ,  $D \rightarrow 111$ ,  
 $E \rightarrow 011$ ,  $F \rightarrow 001$
  - $m$  values require  $\text{ceil}(m/2)$  code bits
  - $n$  code bits can represent  $2^n$  values

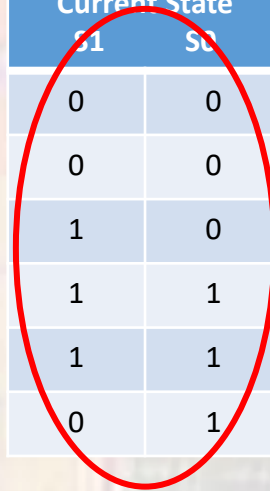


# State Machine Encoding

- Encoding
- Johnson
- Stop light example

Current State S	Inputs		Next State S'
	T <sub>NS</sub>	T <sub>EW</sub>	
ST0	0	x	ST1
ST0	1	x	ST0
ST1	x	x	ST2
ST2	x	0	ST3
ST2	x	1	ST2
ST3	x	x	ST0

Current State S	Current State		Inputs		Next State	
	S1	S0	T <sub>NS</sub>	T <sub>EW</sub>	S1'	S0'
ST0	0	0	0	x	1	0
ST0	0	0	1	x	0	0
ST1	1	0	x	x	1	1
ST2	1	1	x	0	0	1
ST2	1	1	x	1	1	1
ST3	0	1	x	x	0	0



# State Machine Encoding

- Encoding
  - Comparison – considerations
    - Memory elements (bits)  $\rightarrow$  area or fit
    - Coding / Decoding logic  $\rightarrow$  area or fit
    - Speed
      - Shorter logic chains  $\rightarrow$  faster maximum clock speeds
    - Power
      - $P = v * i$
      - $P = v * c * dv/dt$
      - $dv/dt$  represents a transition  $0 \rightarrow 1$
      - Fewer transitions  $\rightarrow$  lower power