Quarter-Step State Transition Table for Standard Quadrature 2-Bit Gray Code Rotary Encoders

Thomas MARTIN, DF7TV, February 12, 2023, Page 1/4

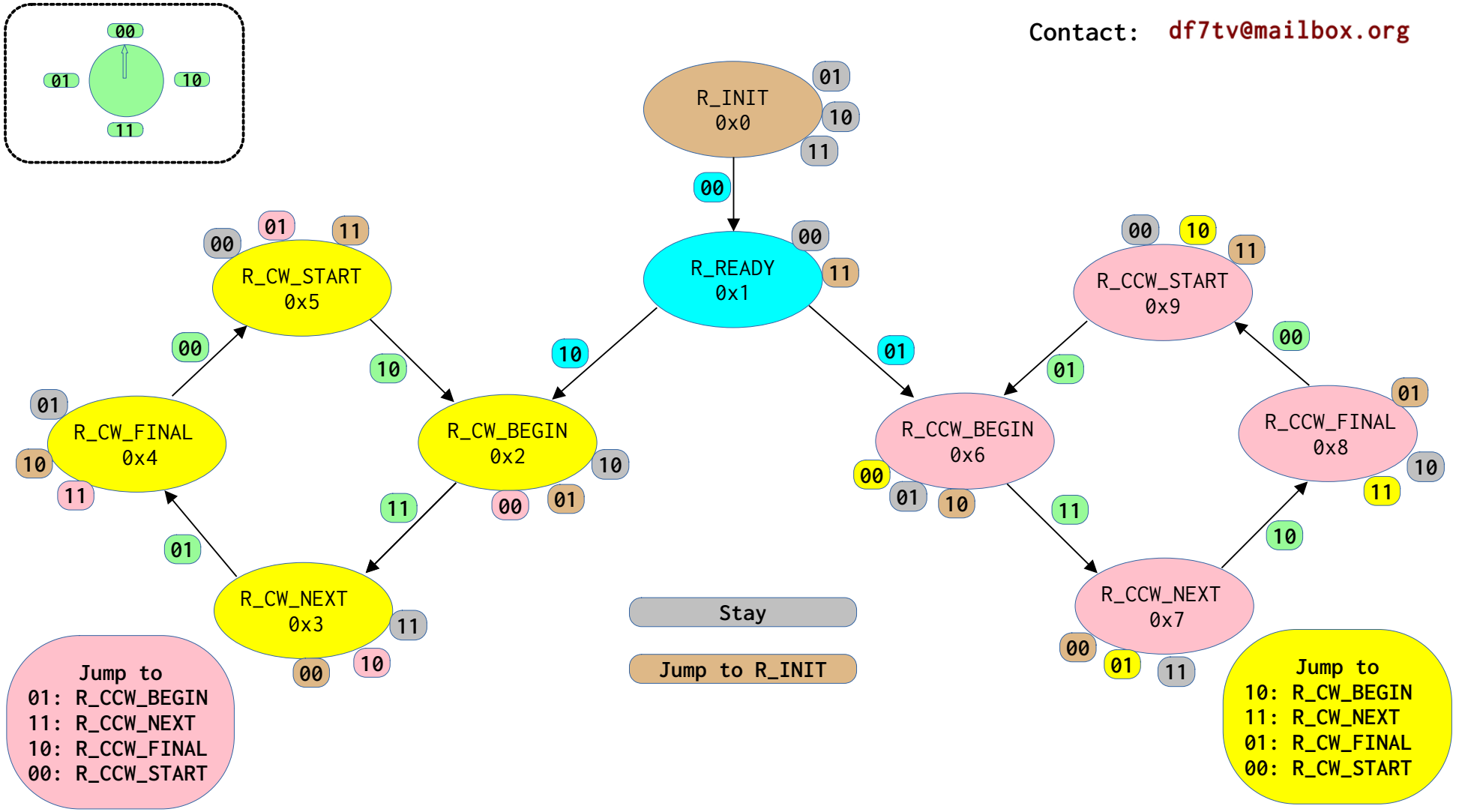

Figure 1: Quarter-Step Mode Finite State Machine (FSM) Diagram

```
// ***************************************************************************
//
// Quarter-Step State Transition Table for Standard Quadrature
// 2-Bit Gray Code Rotary Encoders
// (emits a code at 00, 10, 11 and 01)
//
// Thomas MARTIN, DF7TV, February 12, 2023
//
// Reference:
//
// [1] Buxton, Ben (2011): Rotary Encoder Handler for Arduino, v1.1,
//     https://github.com/buxtronix/arduino/tree/master/libraries/Rotary
//     (Ben Buxton's README.md file describing his Full-Step and Half-Step
//      state transition tables facilitated the development of the
//      new Quarter-Step state transition table)
//
// ***************************************************************************
//
// #define DIR_CW 0x10   // for example
// #define DIR_CCW 0x20  // for example
//
// The Ten States:
//
#define R_INIT 0x0   // Initialization or invalid encoder output; wait for encoder output "00" to jump to state R_READY
#define R_READY 0x1  // Received encoder output "00" when in state R_INIT; now ready to begin
//
// Sense of rotation is clockwise:
#define R_CW_BEGIN 0x2
#define R_CW_NEXT 0x3
#define R_CW_FINAL 0x4
#define R_START_CW 0x5
// Sense of rotation is counter-clockwise:
#define R_CCW_BEGIN 0x6
#define R_CCW_NEXT 0x7
#define R_CCW_FINAL 0x8
#define R_CCW_START 0x9
```

```
// State Transition Table:
//
// Rows represent the current state
// Columns show the transition to the next state due to a new encoder output (sequence 00 - 01 - 10 - 11)
//
// " | DIR_CW" -> emit code "DIR_CW" for a clockwise rotation
// " | DIR_CCW" -> emit code "DIR_CCW" for a counter-clockwise rotation
//
const unsigned char ttable[10][4] = {
//
        //R_INIT 0x0 - Initialization or invalid encoder output; wait for encoder output "00" to jump to state R_READY
   {R_READY, R_INIT, R_INIT, R_INIT},
        //
        //R_READY 0x1 - Received encoder output "00" when in state R_INIT; now ready to begin
   {R_READY, R_CCW_BEGIN, R_CW_BEGIN, R_INIT},
        //
        //R_CW_BEGIN 0x2 - First position of clockwise rotation (Encoder output "10")
   {R_CCW_START, R_INIT, R_CW_BEGIN, R_CW_NEXT | DIR_CW},
        //
        //R_CW_NEXT 0x3 - Second position of clockwise rotation (Encoder output "11")
   {R_INIT, R_CW_FINAL | DIR_CW, R_CCW_FINAL, R_CW_NEXT},
        //
        //R_CW_FINAL 0x4 - Third position of clockwise rotation (Encoder output "01")
   {R_CW_START | DIR_CW, R_CW_FINAL, R_INIT, R_CCW_NEXT},
        //
        //R_CW_START 0x5 - Fourth position of clockwise rotation (Encoder output "00")
   {R_CW_START, R_CCW_BEGIN, R_CW_BEGIN | DIR_CW, R_INIT},
        //
        //R_CCW_BEGIN 0x6 - First position of counter-clockwise rotation (Encoder output "01")
   {R_CW_START, R_CCW_BEGIN, R_INIT, R_CCW_NEXT | DIR_CCW},
        //
        //R_CCW_NEXT 0x7 - Second position of counter-clockwise rotation (Encoder output "11")
   {R_INIT, R_CW_FINAL, R_CCW_FINAL | DIR_CCW, R_CCW_NEXT},
        //
        //R_CCW_FINAL 0x8 - Third position of counter-clockwise rotation (Encoder output "10")
   {R_CCW_START | DIR_CCW, R_INIT, R_CCW_FINAL, R_CW_NEXT},
        //
        //R_CCW_START 0x9 - Fourth position of counter-clockwise rotation  (Encoder output "00")
   {R_CCW_START, R_CCW_BEGIN | DIR_CCW, R_CW_BEGIN, R_INIT}
};
```

State Transition Table for DIR_CW = 0x10 and DIR_CCW = 0x20

```
const unsigned char ttable[10][4] = {
   {0x1,0x0,0x0,0x0},
   {0x1,0x6,0x2,0x0},
   {0x9,0x0,0x2,0x13},
   {0x0,0x14,0x8,0x3},
   {0x15,0x4,0x0,0x7},
   {0x5,0x6,0x12,0x0},
   {0x5,0x6,0x0,0x27},
   {0x0,0x4,0x28,0x7},
   {0x29,0x0,0x8,0x3},
   {0x9,0x26,0x2,0x0}
};
```

State Transition Table for DIR_CW = 0x20 and DIR_CCW = 0x10

```
const unsigned char ttable[10][4] = {
   {0x1,0x0,0x0,0x0},
   {0x1,0x6,0x2,0x0},
   {0x9,0x0,0x2,0x23},
   {0x0,0x24,0x8,0x3},
   {0x25,0x4,0x0,0x7},
   {0x5,0x6,0x22,0x0},
   {0x5,0x6,0x0,0x17},
   {0x0,0x4,0x18,0x7},
   {0x19,0x0,0x8,0x3},
   {0x9,0x16,0x2,0x0}
};
```

License: