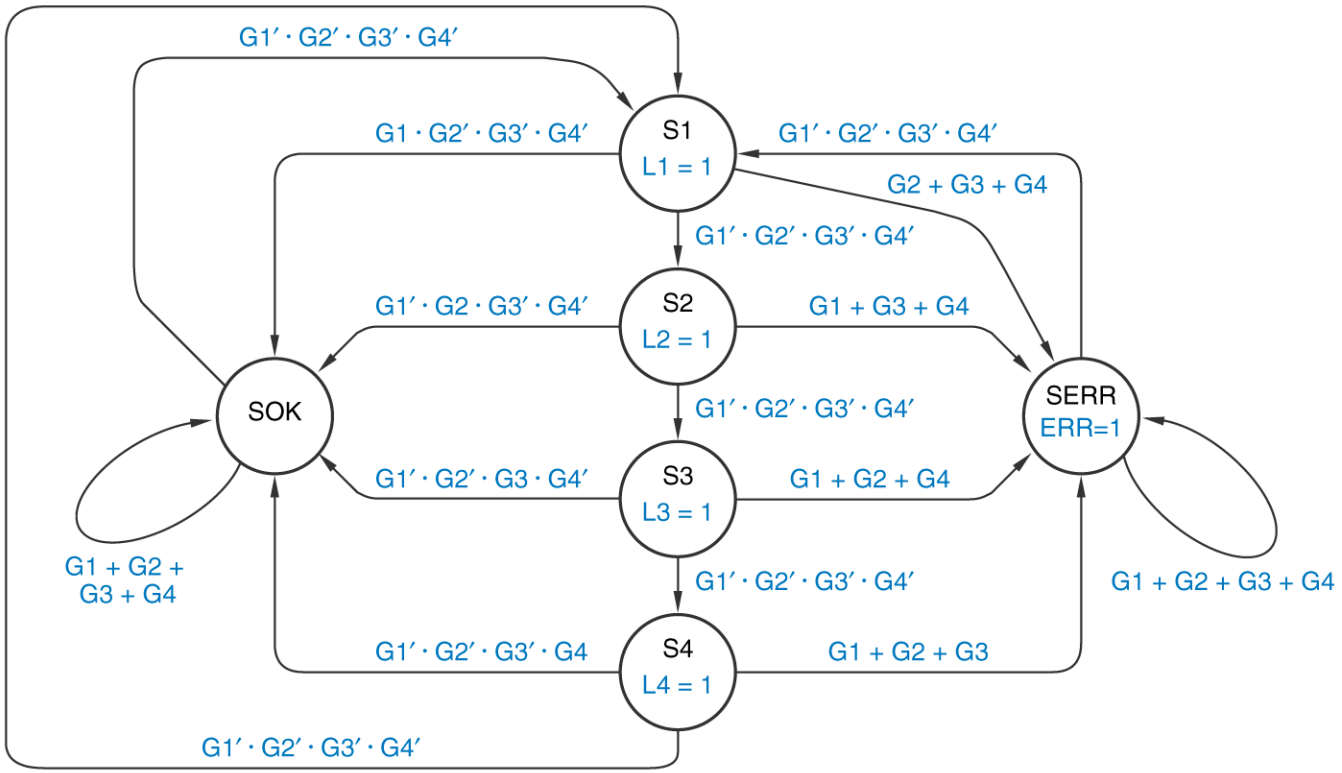

Figure 7-60

Correct state diagram for the guessing game.

PIC16F84A Guessing Game – Assembly Program Instructions

This document explains the functionality and operation of the Guessing Game state-machine program implemented for the PIC16F84A microcontroller. It is written as a user-facing instruction file for anyone reading the code in this repository.

Overview

The assembly program implements a finite state machine (FSM) that cycles through a sequence of four LED states (S1, S2, S3, S4). The user provides a guess through four input lines (G1–G4). A correct guess transitions the system to the WIN state. Any incorrect input transitions the system to the ERR state. After showing WIN or ERR, the system resets back to state S1 and continues cycling. The program runs at a 100 kHz clock frequency and uses a delay routine to generate approximately one second between state changes.

Pin Assignments

Inputs (PORTA, RA0–RA3)

G1 – RA0 – Guess for State 1

G2 – RA1 – Guess for State 2

G3 – RA2 – Guess for State 3

G4 – RA3 – Guess for State 4

Inputs are active-low: when a guess is pressed, the RA pin reads 0.

Outputs (PORTB, RB0–RB5)

L1 – RB0 – State S1 indicator

L2 – RB1 – State S2 indicator

L3 – RB2 – State S3 indicator

L4 – RB3 – State S4 indicator

ERR – RB4 – Incorrect guess indicator

WIN – RB5 – Correct guess indicator

The program uses one-hot encoding so only one output bit is active at a time.

State Machine Behavior

The program begins in State S1, sets the corresponding LED, waits approximately one second, and transitions to the next state in sequence.

State Cycle:

S1 → S2 → S3 → S4 → S1 → ...

At each state, the program checks inputs:

- If the correct input for that state is low, the machine transitions to WIN.

- If any other input is low, it transitions to ERR.

- If no input is pressed, it proceeds to the next state.

WIN State

When the correct guess is detected, the WIN output is activated. The system waits approximately one second while in the WIN state. As long as any input remains active, the system stays in WIN. When all inputs are released, the system resets to S1.

ERR State

When an incorrect guess is detected, the ERR output is activated. The system holds this state until no input is active. It then resets to S1 and resumes normal cycling.

Delay Routine

A two-level nested loop generates a delay of approximately 986 milliseconds at a 100 kHz clock frequency. This delay controls:

- State transitions

- WIN display duration

- ERR display duration

Program Architecture

1. Initialization

- Configures PORTB outputs
- Loads initial state (S1)
- Outputs the initial state on PORTB
- Delays and enters the state cycle

2. State Handlers (State1, State2, State3, State4)

- Evaluate input conditions
- Branch to WIN or ERR if needed
- Update next state
- Output state to PORTB
- Delay and proceed

3. WIN and ERR Handling

- Indicate result through LED output
- Wait until inputs are cleared
- Reset to S1

4. Delay Subroutine

- Creates the timing loop for state transitions

What This Program Demonstrates

The implementation demonstrates skill in:

- Embedded assembly programming
- Finite state machine construction
- GPIO control and register-level configuration
- Timing loops and clock-cycle analysis

- Input-based branching logic

- Microcontroller simulation and debugging

Files Included

guessing_game.asm – Assembly source code

INSTRUCTIONS.pdf – Explanation of system behavior and program structure