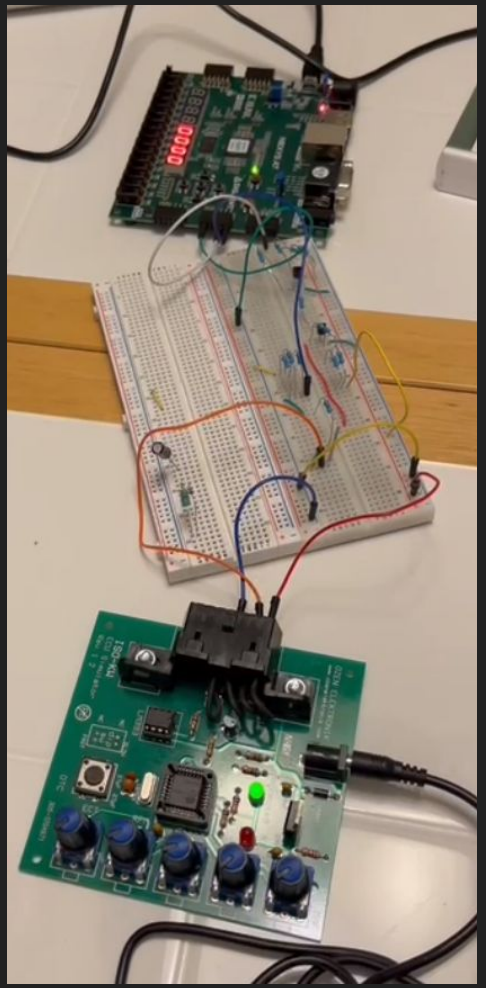
K-Line Communication

Group Members: Jacob Alam, Adam Jesse, Trey Plichta and Ruger Stellberger

Introduction:

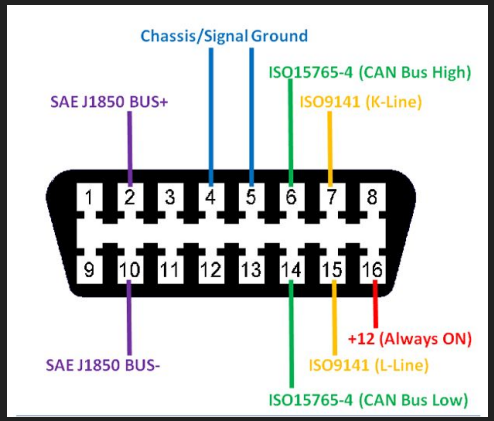
- The Purpose of this project is to establish K-line communication between a Nexys-A7 FPGA and a ISO 9141 OBD ECU simulator board.
- Hardware components necessary for this project include: Nexys-A7 FPGA, ISO 9141 OBD ECU simulator board, 12V to 3.3V down converter circuit, and OBD-2 connector.



What is K-line Communication?

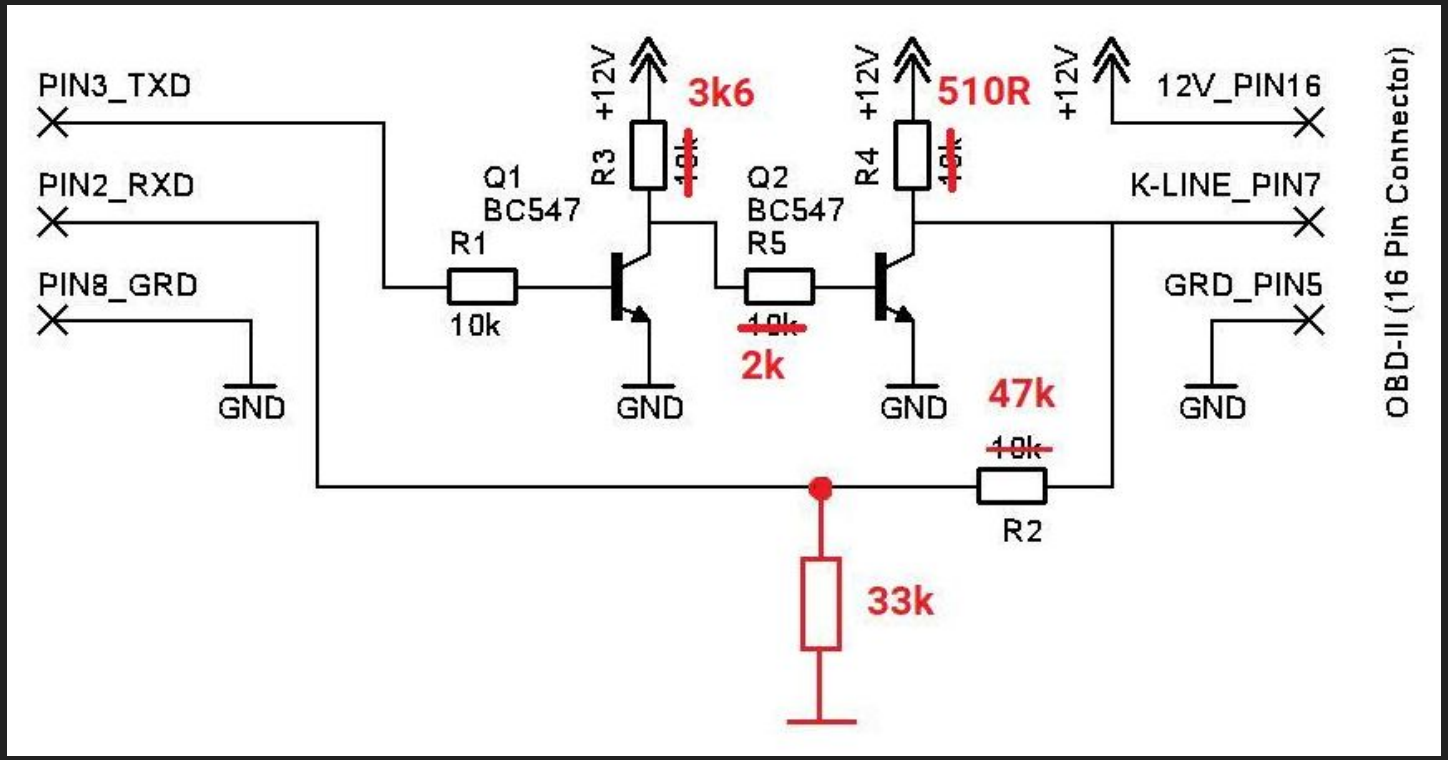
- K-line communication is a single wire communication protocol that allows many components to communicate via encrypted data using PWM.
- It's main use is in automobiles to transmit electronic diagnostic signals between Electronic Control Modules (ECMs) and diagnostic equipment.

https://www.picoauto.com/library/automotive-guided-tests/k-line



https://components101.com/connectors/obd2

Circuit Diagram



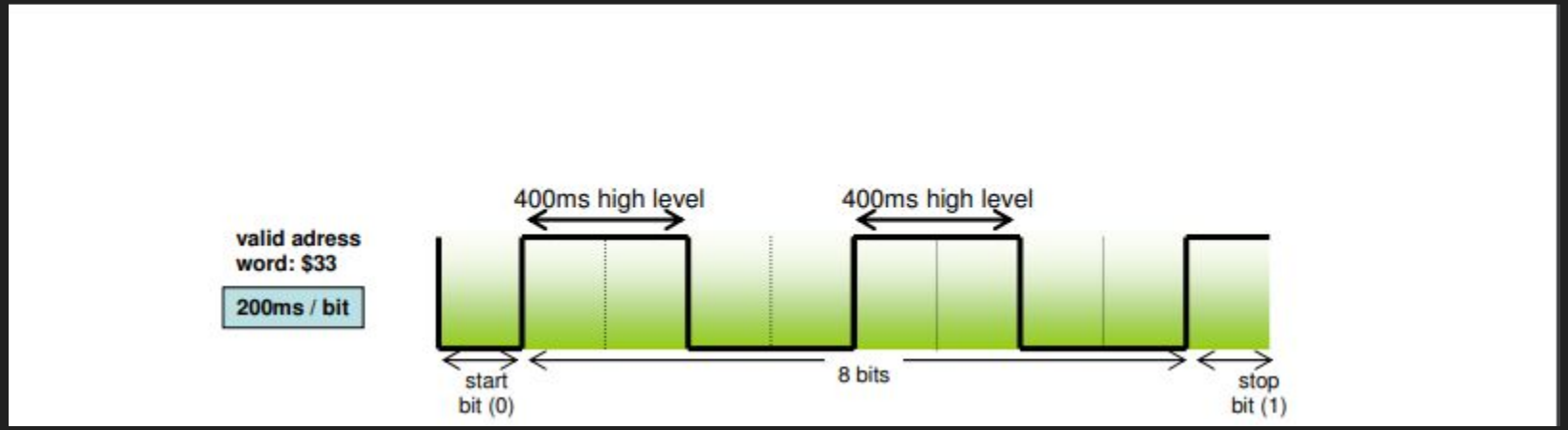
https://forum.arduino.cc/t/comunicating-uno-iso9141-2/643216/8

Initialization Process

The steps to initialize the communication between the FPGA and emulator:

- Nexys issues 0x33 to emulator over 2 second period
- Emulator sends 0x55 to issue to Nexys Baud rate must change to 10.4kbps.
- Emulator waits between 5ms and 20ms for Nexys to configure baud.
- Emulator will now issue key bytes, separated by a time between 0 and 20ms, that entail either 08 08 or 94 94.
- The Nexys will now wait between 25 and 50 ms to invert byte #2 and send it back to the vehicle
- Emulator will now invert byte 0x33 and send it back to the Nexys completing the process.

Initialization Process



Components Used in Initialization Process

Counters:

- Used to wait for messages and send data at different Baud rates.

Right Shift Registers:

- Used to shift out data using Pmod headers to Emulator board.

Falling Edge Detector:

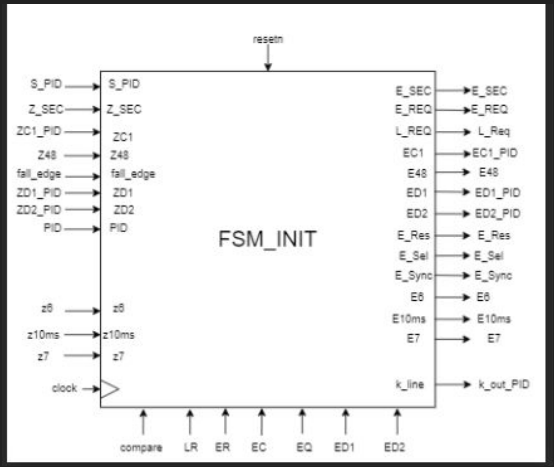
- Allows the circuit to determine the status of a received payload.

Register and Comparator:

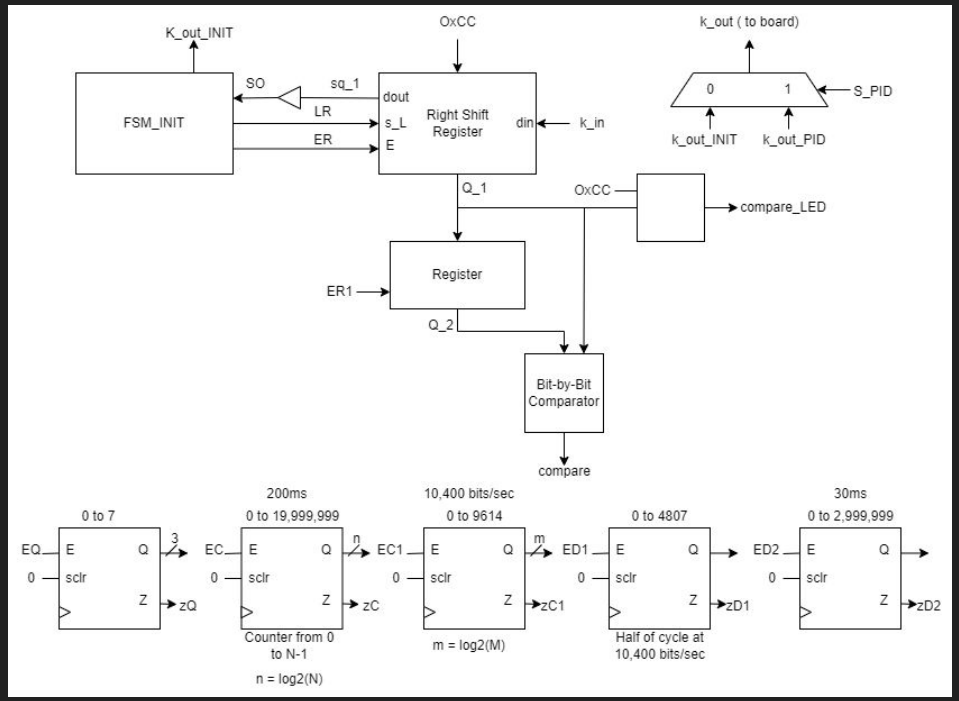
- Debugging: Used to verify correct bits being sent to Emulator.

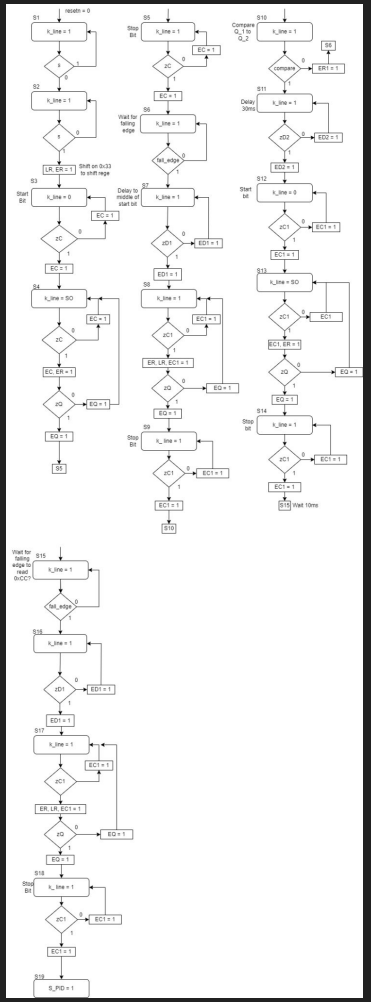
Init State Machine

- The Init State Machine is a state machine with 19 states. The purpose of this process is to complete each step of the initialization process in its required time and sequence.
- Some states depend on a reading for the counter components to advance to the next state. This allows the circuit to wait for the emulator wait times and to allow the FPGA to send out data only when necessary.



Initialization Circuit Diagram:





Reading PIDs from Emulator Board

- Once the Emulator board was initialized using the process previously stated, the system was now ready to receive and read PIDs.
- The PIDs signify Parameter Identification codes that the Nexys will receive from the Emulator board.
- In this segment, the Nexys board was used to receive the PIDs and display them using the built in seven segment displays.
- PIDs require a baud rate of 10400 bps.
- The Data format is as follows:

Format	Target	Source	DLC	Data	Checksum

Components used in Reading PIDs

Counters:

- Allows for correct timing when receiving and transmitting signals.

Multiplexor:

- Selects between different signals to be transmitted.

Right Shift Register:

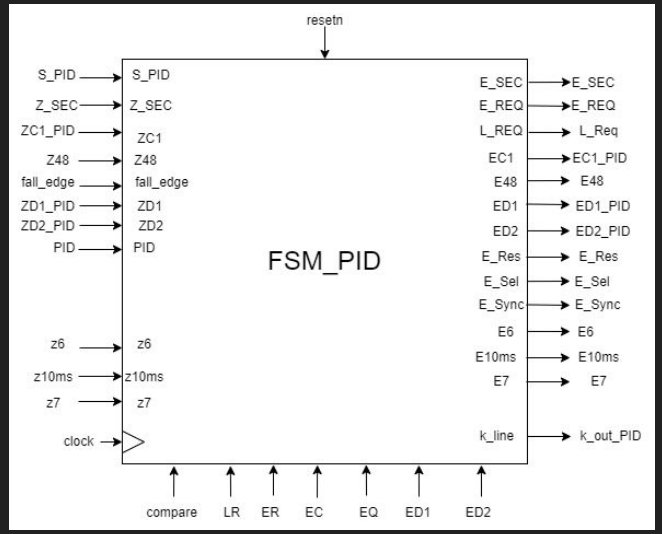
- Shifts data to output pin.

Register:

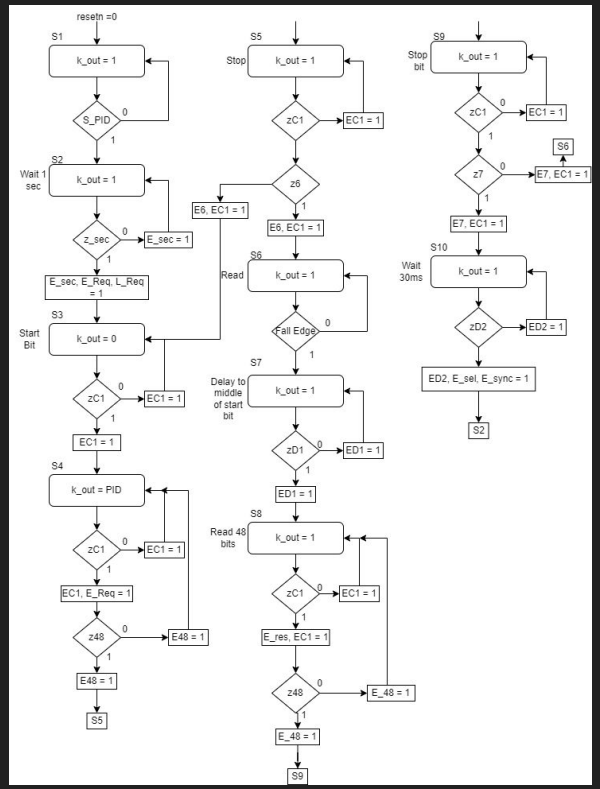
- Holds data received from Emulator board.

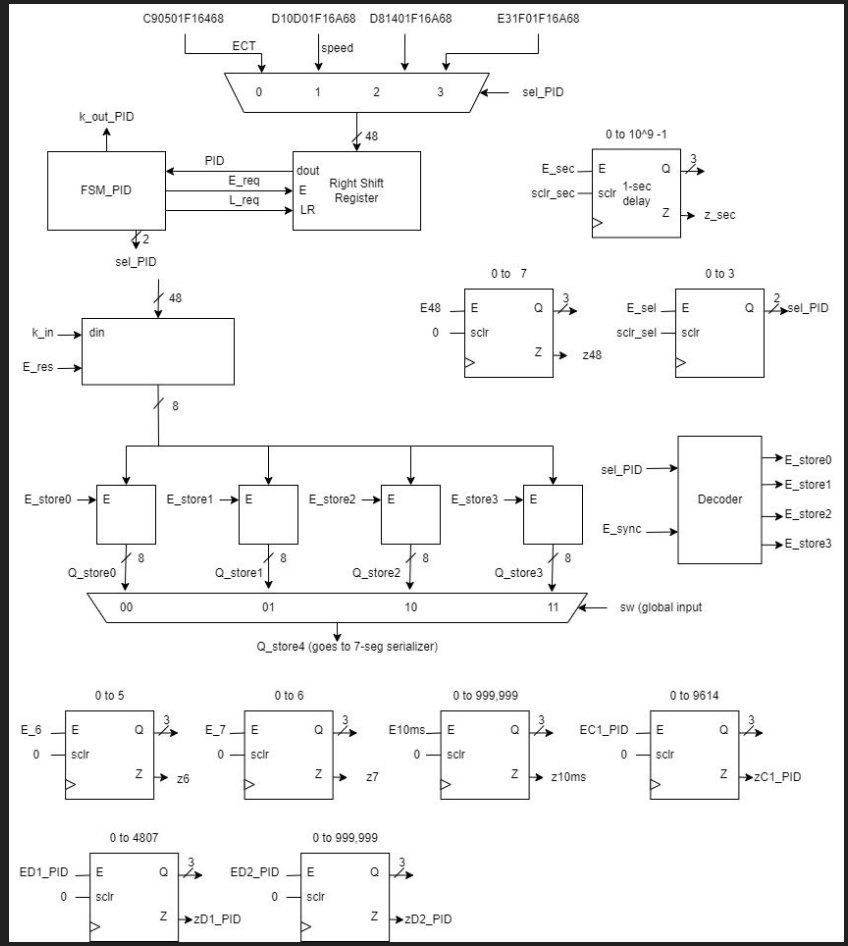
PID State Machine

- This component functions in similar fashion to the initialization state machine.
- This FSM contains 10 states that allow the circuit to recognize and display PIDs.
- The PID state machine directly signals other components such as counters, registers, multiplexers and right shift registers.
- The PID state machine directly takes the inputs from counters to allow it to wait designated times when reading PIDs.
- The state machine also regulates the bits shifted out of the Pmod headers connected to the emulator board.



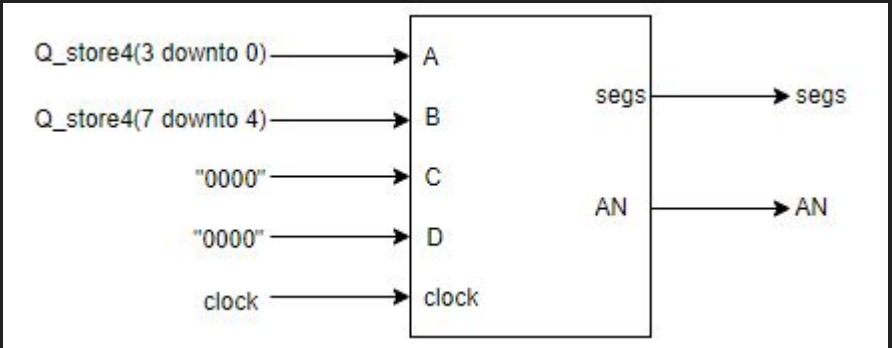
PID Circuit Diagram:



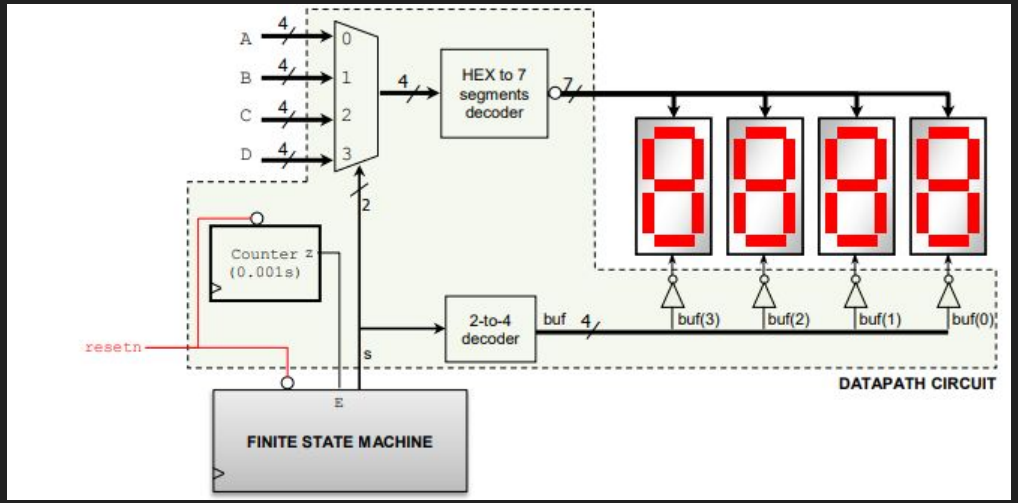


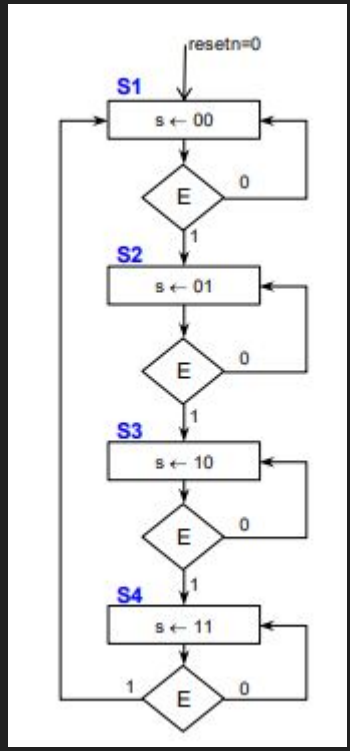
Seven Segment Serializer

- This component was used to receive signals from the PID state machine and display the PIDs emitted from the emulator board.
- The PIDs are displayed in hexadecimal format.
- A multiplexor controls what PID is displayed on the seven segment displays using PIDs stored in registers and a 2 bit select controlled by switch 0 and 1 on the Nexys board.



Seven Segment Serializer Diagrams

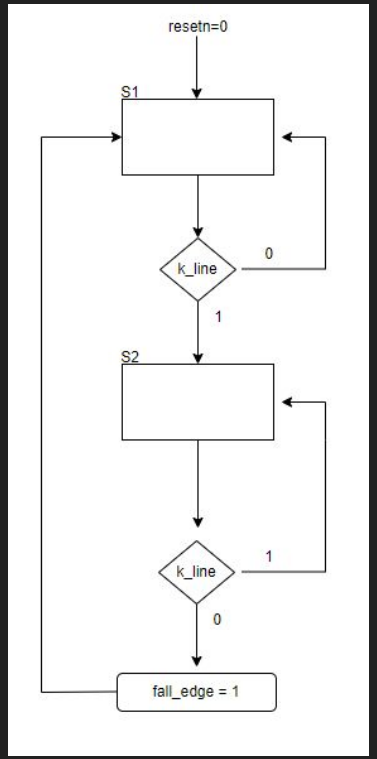




Note: These components were not a focal point of the project. Therefore, VHDL code and diagrams were taken directly from notes and online resource website.

Falling Edge Detector

Both the Initialization FSM and the PID FSM utilized a common falling edge detector circuit in order to begin reading data from the emulator board. After receiving an alert from the Falling Edge Detector circuits, these control FSMs could proceed to the next state.



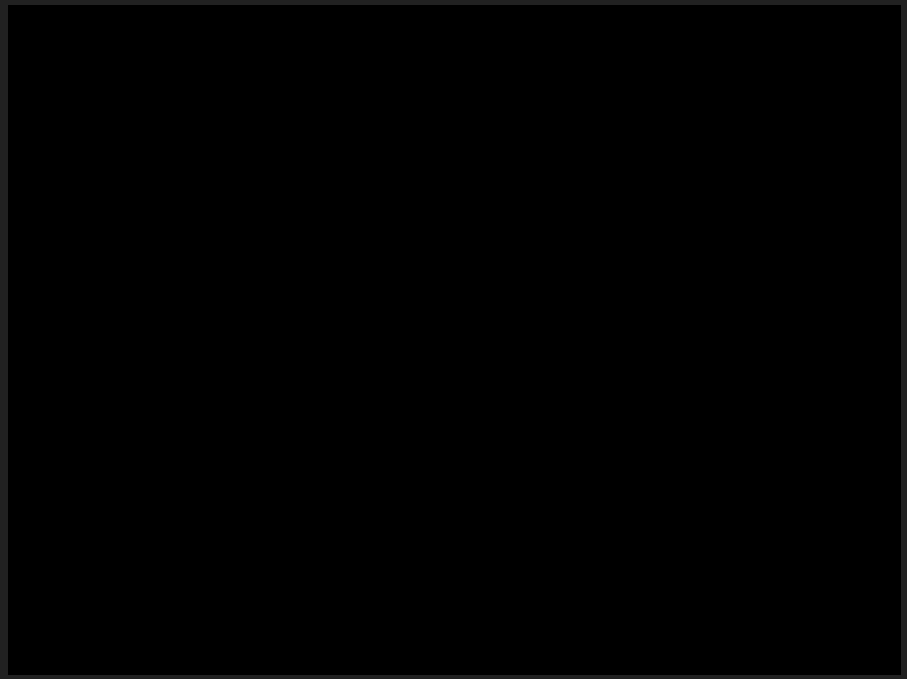
Inputs

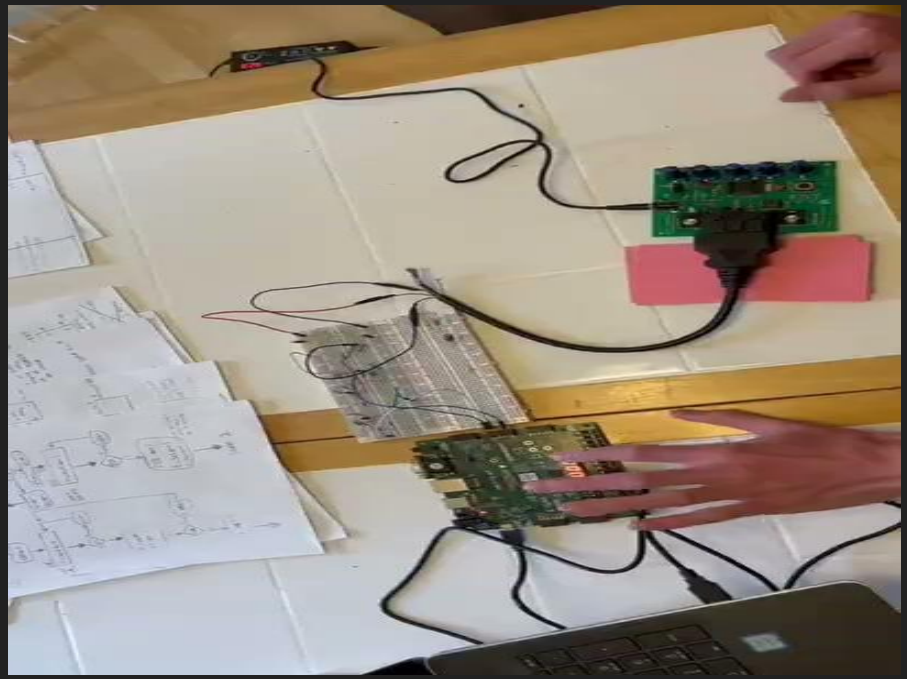
clock, resetn, k_in (k_line within the falling edge detector)

Outputs

fall_edge

Video





Live DEMO

References:

[1] B. Gruszczynski, “K-line Communication Description,” Nov. 2009. Accessed: Apr. 12, 2023. [Online]. Available: https://www.obdclearinghouse.com/Files/viewFile?fileID=1380

[2] Llamocca, Daniel. VHDL Coding for Fpgas, Oakland University. https://www.secs.oakland.edu/~llamocca/VHDLforFPGAs.html.