WP-1001 Asynchronous State Machine vs MCU

Asynchronous state machines can take on a variety of embedded control applications that would formerly have been the exclusive domain of microcontrollers. The easily configured asynchronous state machines brings key advantages of ultra-fast state transitions, leakage level static current consumption, robust design, and supply voltage tolerance so important in IoT, portable, mobile and wearable applications.

When designing portable battery powered systems, size and battery life constraints are often the most severe challenges to overcome. One traditional way to pack a lot of functionality into a small size and power budget is to use a low power microcontroller such as TI's popular MSP430. These types of ultra-low power microcontrollers offer high levels of flexibility and are available in small packages, allowing a system to run for not only days, but months and years on a single battery cell. Silego Technology has taken a different approach in attacking this same problem, by adding a user programmable Asynchronous State Machine macrocell to our fifth generation GreenPAK™ product family.

This white paper contains comparisons, design tradeoffs, and tips that the user may consider when choosing between a variety of microcontroller options, and doing the same job using the asynchronous state machine inside Silego's GreenPAK.

How much MCU code can the GreenPAK 5 handle?

GPAK 5's ASM contains 8 states and 24 possible decisions.

To help the reader grasp the ASM concept quickly perhaps this will help –

- STATE(X): If Signal One is high, then go to STATE(A). *This is a decision statement to determine the next state; each state is allowed to be the resultant state only three times.*
- State A is a configuration that causes an output to blink an LED. *This is the actual result*. *In order to move to the next state (result), another decision must be made.*
- STATE(A):
  - If Signal Two is high, then go to STATE(B)
  - If Signal Three is high, then go to STATE(D)
  - If Signal Four is high, then go to STATE(F)
- STATE(B) might cause an LED to blink faster
- STATE(D) might wait 500ms before moving on to another STATE
- STATE(F) might check the POR or a Power Good signal.

So the ASM in PAK5 represents an MCU program with up to 24 IF..THEN statements. When the 8 State ASM capabilities are considered together with hardware input and output circuits, the GPAK 5 may be represented as being roughly equivalent to about 100 lines of standard C code written for common 8 and 16 bit MCUs.

As the name implies ASMs have no clock. Silego's GPAK 5 ASM is event driven and does not operate with a clock. As such, when there are no events, the ASM stays in one state and consumes no static power.

As a result, applications with limited input cycles can operate at leakage current power consumption well in to the single digit nanoamps of average current consumption at room temperature. Applications such as resets, power sequencing, various sensor interfaces, longterm data logging may benefit from a real ASM.

What types of embedded control problems are solved with GreenPAK 5's ASM?

Typical embedded control problems usually involve a system that is transitioning through a set of discrete states based on asynchronous external inputs. ASMs are the natural problem solver for most embedded control problems. However, most designers opted for traditional microcontrollers resulting in asynchronous state machines being pushed to the edges of engineering. However, due to inherent power and speed advantages of ASMs the on-going low-power portable consumer/wearable/IoT revolution has renewed interest in ASMs. Silego has revived and modernized the ASM, mitigating the well-known hazard and race conditions, the programming/configuration headaches, while retaining all the inherent low-power, lowlatency benefits for simple (up to 8 states) embedded control problems that would require less than 100 lines of code.

The types of embedded control problems that can be solved by the GPAK 5 ASM are limited by

a. The number of inputs: 18

b. The combinatorial logic blocks processing the 18 inputs: 1

c. The number of total states: 8

d.The number of decisions: 24

e. Resolving basic true / false decisions from combinatorial logic

1) Is the voltage higher or lower than X volts?

2) Is Signal A XOR Signal B = 0 or 1?

3) Has timer Z finished?

f. 4 analog comparators

g. 8 to 5.5V VDD

h. ~25 MHz digital signals

Silego's GPAK 5 has almost no computational power and is therefore not suitable for digital filtering, solving multi-bit math problems, graphics, vector calculations, etc. Silego's GPAK 5 has only 8 bytes of I2C addressable memory and is in most cases not suitable for an interface converter.

An intelligent flashlight is an easy to understand example of how the GPAK 5 ASM could be used. This flashlight has an OFF state, a High Brightness State, a Low Brightness Battery saving State, A flashing State, a Battery Status State. The GPAK 5 ASM can easily be set up to cycle between these states perhaps with button pushes or multiple button inputs. The output drive MOSFET can be controlled by the GPAK 5 to achieve high steady current through the White LED or lower current or flashing current depending on the configuration of the combinatorial logic as activated by the ASM's 8 output bits.

IO port expansion is another easy to understand application example where GPAK 5 can be found useful. GPAK 5 comes with an I2C port that can be used to directly drive outputs, the ASM, or combinatorial logic. As mentioned, the I2C can write directly to a pin. However, the I2C bus can also be used indirectly as a smart I2C port expander. For example, an I2C byte can be used as an input into the ASM. The configured ASM will respond to these inputs changing states as desired.

Each state of the ASM is mapped to 8 configurable bits that may be connected to GPAK 5 outputs. Thus a single bit in an I2C byte could be mapped to the bits necessary to drive a "0" in a seven segment display. The next state could be mapped to drive a "1", and so on. IO port expansion can be achieved even without using I2C. For example, a single pin can be pulsed to transition ASM between states and thus cause up to 8 pins to have state dependent pre-configured outputs.

Overkill Microcontrollers vs. GPAK 5 ASM Value

Microcontrollers contain a processor, program code, stack memory, and various peripherals. Microcontrollers can easily perform the application examples above but are inefficient in size and power. It is quite common to find MCUs designed into applications where less than 1% of the MCU horsepower will ever be used.

GPAK 5's ASM is well suited to simple embedded control applications, especially ultra-low power applications. In fact, most MCUs are computational power and cost overkill for these simple embedded control applications making Silego's Low Cost GPAK 5 ASM, a No Code, No Static Power, Ultra Small-size, easy-to-use option.

Interrupt Latency (ns vs. us)

State machine design on a microcontroller is done in software running on the microcontroller core.

In this case, the states are implemented as points in the software instruction execution, and state transitions are implemented with conditional software branching. Microcontrollers have the capability to also process asynchronous inputs, and they do this through dedicated interrupt controller hardware, and through interrupt service routines (ISRs).

The ISR is software that is run after a hardware interrupt has been activated. An important benchmark for microcontrollers is how short is the time from an external interrupt signal until the core is executing the first instruction of the ISR, the so-called interrupt latency. MCU interrupt latency in general purpose devices is usually measured in fast examples at around 5 to 10us.

Silego's ASM equivalent of interrupt latency is measured in nanoseconds – equivalent to the unclocked time of flight through the few gates between an external pin and the internal ASM input. The ASM has latency from one state to the next. If the GPAK 5 is operating at 5V power supply, the latency is a maximum of 50ns. Yes, ASMs are extremely fast and extremely low power. Even worse, MCU interrupt latency includes a variety of tasks that the microcontroller must do before it can execute this first instruction, including latching the external signal, finishing execution on the instruction currently in progress and saving state (pushing registers) to be able to return to exactly the same state. Depending on the particular microcontroller, the latency measured in cycles can be as small as 10-20. Multiplying these cycles by the clock speed yields the interrupt latency.

There is also an impact from running a high level language (such as C), as this usually increases the number of cycles for the interrupt latency. Similarly, if the user is running an operating system, this can also add extra cycles to the interrupt latency.

There is a tradeoff that the microcontroller user must make if power and fast interrupt response are both required in the system: the faster the clock speed is set, the lower the interrupt latency will be, but a faster clock speed will also drive up the power consumption. Some microcontrollers allow the system clock speed to change over time, so that the system clock speed can be set low when the system is waiting on an asynchronous input, and increased when the interrupt is requested. This can be a method to save power and decrease interrupt latency, but there is a limit to how useful this is, as there is also latency in changing the clock speed. No such tradeoff is necessary with an ASM.

VDD variation

Silego's ASM works over a wide voltage range. A properly designed ASM is guaranteed hazard and race condition free because each ASM signal path is guaranteed by signal length and gate count. Thus as the VDD changes, so does the propagation delay. Further, the propagation delays are all matched and thus performance is guaranteed.

Microcontrollers, on the other hand, are clocked with signals that are not correlated well with VDD.

As the VDD changes, the MCU propagation delays change and since the timing doesn't change, the timing margins are soon compromised.

Chip designs react to these design hazards but putting the MCU under a voltage regulator or requiring still more performance to be lost by slowing the clock speed. However, the voltage regulator consumes power, and the slower clock speed increases interrupt latency.

Crash vs. No Crash

There are design and system flaws that can cause a microcontroller to crash. Poor software design, timing issues, miscalculations of interrupt latency, running out of stack memory, memory leakages, and accidental writes in program memory are some common pitfalls that cause MCUs to crash./span>

Silego's ASM is configured in hardware with NVM bits, has no timing issues, latency measured in ns, no stack memory, no ability for memory leakage, and no ability to unintentionally overwrite program memory, and is therefore inherently more robust with VDD noise and brownouts./span>

Thus it is common to find Silego's GPAK ICs and GPAK 5's ASM being issued has crash monitors for large MCUs, SOCs, and ASICs. Some are also used to operate the failure mode notifications, error LEDs, and other functions that must remain operational while the main processor is malfunctioning.

No Code GUI based tools vs. typical MCU tools

Silego's ASM is configured using the GreenPAK Designer development environment. The software appears to be a schematic capture editor instead of a coding tool. Most state machine designs can be implemented in a matter of minutes reducing the typical MCU tool learning curve from months to Silego's GPAK learning curve of a few days.

Inside this tool, a special bubble state diagram tools allows the designer to select and name the various states in the system, point and click to add state transition arrows in the same manner that an engineer would draw a state diagram on a white board.

After configuring the state machine the designer returns to the schematic capture editor to graphically connect the hardware signals to each of the state transition decision points. It is common for a design engineer to be able to complete their first prototype in one day of effort.

GPAK 5 size vs. Low Power MCUs

Without all the complexity of the MCU architecture the GPAK 5 is usually smaller, especially when compared to 1.8 to 5V tolerant, low-pin count, ultra-low power MCUs. If a GPAK 5 product is able to perform the control function, then it is also usually the smallest programmable option on the market at 2 x 3 mm 20-pin STQFN.

For applications that require less control; previous GPAK family are offered in packages as small as 1.6 x 1.6 mm 12-pin STQFN.

Conclusion

In summary, the tiny GPAK 5 with the 8 state ASM can take on a variety of embedded control applications that would formerly have been the exclusive domain of microcontrollers. The easily configured ASM brings key advantages of ultra-fast state transitions, leakage level static current consumption, robust design, and supply voltage tolerance so important in IoT, portable, mobile and wearable applications.

More information

GreenPAK Landing Page

Datasheets

Development Software

Contact: appnotes@silego.com

Files

- WP-1001 Asynchronous State Machine vs MCU.pdf- (66 KB)
- WP-1001.zip- (44 KB)

See full list of Application Notes