arm Education

Embedded Systems Essentials with Arm: Getting Started

Module 4

TP (4): Thinking point

Imagine you work in a small and slightly chaotic company making and selling children's toys. There's a number of things you know you need to do this morning. You sit down at your desk and start getting on with them, in your usual methodical way.

Just as you're getting stuck in, your supervisor comes past. Could you just look at this little problem with a difficult customer? It shouldn't take long. You sigh inwardly, and write a scribbled note of where you've reached in your own set of tasks.

A short while later, you're just beginning to work out what went wrong with this customer, when the company boss comes rushing in. He's a smart guy, full of the best ideas, but organization is not his strong point. There's a big issue with a supplier for one of the company's most popular products. Finance forgot to pay them, and he needs you to give them a call. You sigh inwardly yet again, and make another scribbled note as to where you've reached with your supervisor's task.

You were getting somewhere with that, but what the boss needs pushes everything else aside. You call the supplier. You use your sweetest telephone style to calm them down and promise immediate payment. While you're on the phone, someone comes by your desk collecting for a leaving present. You signal in the nicest way you can for them to come back later. You put the phone down and email the boss. Then you return to what your supervisor wanted, finish that, and half an hour later you're back to your own task list. You put on your big headphones – nobody can interrupt you now!

You're getting along famously when there's a harsh clanging sound in your ears, even the headphones won't keep that out! It's the fire alarm. Calmly you leave your desk and everything that's on it, as the fire drill has taught you, and head out to the safe zone in the company car park. You find out there's been a small fire in the packing department, but it's been put out already. What a morning it's been!

In this little scenario you were systematically getting on with your routine tasks, but had to deal with interruptions along the way. The life of a microcontroller in an embedded system is very similar. It is likely to have a program with a set of routine tasks, which it starts executing in a pre-planned sequence. But there are important mechanisms which allow interruptions. Such inputs, coming from a peripheral or the outside world, can interrupt the CPU in whatever it's doing, and tell it to do something else. And if one interrupt is more important than another, it can claim precedence - think of the company boss's task taking over from your supervisor's. When the interrupt has been sorted, the CPU can get back to its regular tasks. Just like you, the CPU needs to keep note of where it was, and what it was doing.

If the programmer feels it's necessary, the CPU can block out many interrupts, just like when you put on your headphones. Most interrupts will contribute to ensuring the overall system performs as it is expected. However, a small number of interrupts, like the fire alarm in our imaginary scenario, could relate to system survival and will be top priority. These, normally, can *never* be blocked out.

Dealing with Interrupts becomes a hugely important part of embedded system programming, and that's what this module is all about.

Think of a time in the past few weeks when you have been trying to get something done, but have been repeatedly interrupted. Possibly you got quite annoyed! Was your response to one of those interrupts then itself interrupted, by something more important? How did you deal with it? Did you do anything to remind yourself where you were, when you had the chance to get back to your first task? Enjoy learning about Interrupts!