The Device Driver

CM0506 – Small Embedded Systems

Dr Alun Moon

Department of Computer and Information Science

Lecture 3

Interpretation of Hardware Specifications

Typical hardware specifications include:

- Functional description
- Pinout specifications
- Operating voltages (Minimum, maximum, and typical)
- Timing Diagrams
- Protocol Diagrams
- Critical timing data

What is a device driver?

- A collection of software routines to perform I/O functions
- Interface software, called by the operating system or application code, to configure devices and perform I/O
- Software to *glue* the hardware and software together
- Separates policy from mechanism

A Device Driver

- Encapsulates the behaviour of a device
- Allows application developers to ignore low-level detail
- A consistent interface to a device or family of devices

Device Driver code

- Notoriously difficult to design and debug
- May be complex
- Requires a deep understanding of the hardware
- Low-level code – sometimes requires assembly language
- API (Application Programming Interface) requires careful design

Portability

- Device drivers provide a layer of abstraction to hardware I/O devices
- Higher levels of software can access devices in a uniform hardware-independent manner
- If designed well, device driver software can be ported.

Developing Device Drivers

- Read the hardware specification
- Re-read the specification, review in a group
- Specify an API and review this
- Design and develop code to provide the API and consistent with hardware specifications
- Test the API carefully – use instumentation, and simple, incremental, text harness software.

Typical Driver Functions

- Configure a device – initialise the hardware to a known state
- Turn a device on or off
- Assign interrupt handlers
- Read data from a device
- Write data to a device

Timing diagrams

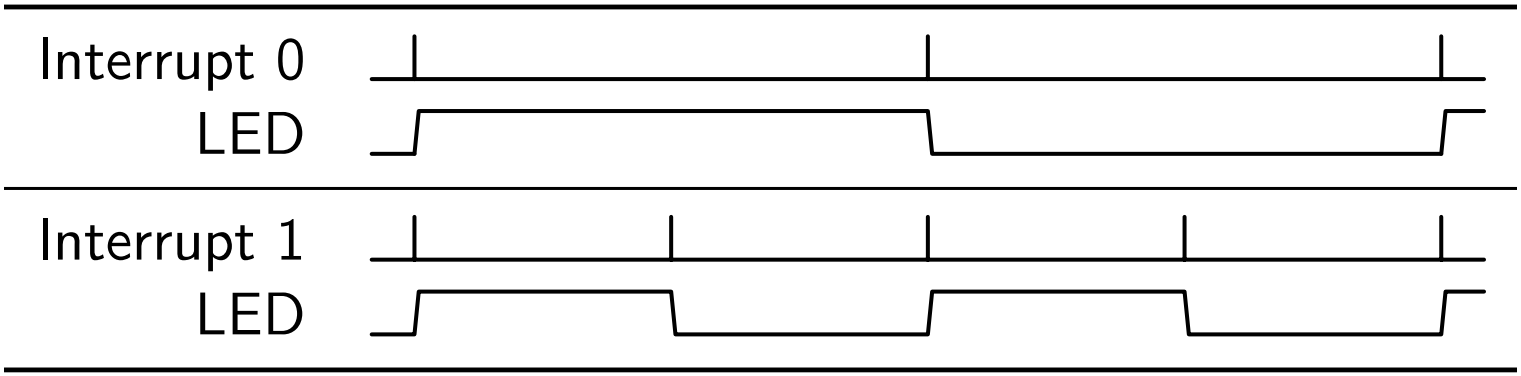

Table: Timing diagram for two LEDs