Resource R11 Interrupt functions documentation

This resource includes some documentation of functions from the National Instruments `C_Support_for_myRIO` library (included in the `me477` library) used in Lab Exercise 05. For more details, see the `me477` library header files `DIIRQ.h` and `IRQConfigure.h` and POSIX C library `pthread.h`.

Resource R11.6 Register DI IRQ

`Irq_RegisterDiIrq()` Reserves the interrupt from FPGA and configures DI IRQ. Declared in the `DIIRQ.h` header file.

Prototype:

```
int32_t Irq_RegisterDiIrq(
  MyRio_IrqDi       *irqChannel,
  NiFpga_IrqContext *irqContext,
  uint8_t           irqNumber,
  uint32_t          count,
  Irq_Dio_Type      type
);
```

Arguments:

- `irqChannel` structure containing the registers and settings for a digital IRQ I/O
- `irqContext` IRQ context to be reserved
- `irqNumber` the IRQ number (`IRQNO_MIN`–`IRQNO_MAX`)
- `count` the incremental times that you use to trigger the interrupt
- `type` the trigger type that you use to increment the count
- **`return`** the configuration status

Resource R11.7 Unregister DI IRQ

`Irq_UnregisterDiIrq()` Clears the DI IRQ configuration setting. Declared in the `DIIRQ.h` header file.

Prototype:

```
int32_t Irq_UnregisterDiIrq(
  MyRio_IrqDi       *irqChannel,
  NiFpga_IrqContext irqContext,
```

```
  uint8_t           irqNumber
);
```

Arguments:

- *irqChannel structure containing the registers and settings for a digital IRQ I/O
- irqContext IRQ context to be reserved
- irqNumber the IRQ number (IRQNO_MIN–IRQNO_MAX)

Resource R11.8 Wait for Interrupt

Irq_Wait()	Wait until the specified IRQ number occurred or ready signal arrives. Declared in the IRQConfigure.h header file.

Prototype:

```
void Irq_Wait(
  NiFpga_IrqContext irqContext,
  NiFpga_Irq        irqNumber,
  uint32_t          *irqAssert,
  NiFpga_Bool       *continueWaiting
);
```

Arguments:

- irqContext context of current IRQ
- irqNumber IRQ number
- continueWaiting signal which aborts the waiting thread
- **return** irqAssert asserted IRQ number

This is a blocking function that stops the calling thread until the FPGA asserts any IRQ in the number parameter, or until the function call times out. The irqsAssert parameter can be used to determine which IRQs were asserted for each function call.

Resource R11.9 Acknowledge IRQ

Irq_Acknowledge()	Acknowledges an IRQ to the FPGA. Declared in the IRQConfigure.h header file.

Prototype:

```
void Irq_Acknowledge(
  uint32_t irqAssert
);
```

Arguments:

- `irqAssert` asserted IRQ number

Resource R11.10 Create POSIX thread

`pthread_create()` Creates a new thread within a process. Declared in the `pthread.h` header file.

Prototype:

```
int pthread_create(
  pthread_t            *thread,
  const pthread_attr_t *attr,
  void                 *(*start_routine) (void *),
  void                 *arg
);
```

Arguments:

- `*thread` new thread identifier
- `*attr` new thread attributes (`NULL` - default)
- `*start_routine` starting function of new thread
- `*arg` sole argument of `start_routine`
- `return status = 0` for success

Resource R11.11 Join POSIX thread

`pthread_join()` Suspends execution of the calling thread until the target thread terminates. Declared in the `pthread.h` header file.

Prototype:

```
int pthread_join(
  pthread_t thread,
  void      **retval
);
```

Arguments:

- **thread** thread identifier
- *retval if not NULL, copies the exit status into the location pointed to by retval
- **return** status = 0 for success

Resource R11.12 Exit POSIX thread

`pthread_exit()` Terminates the calling thread. Declared in the `pthread.h` header file.

Prototype:

```
void pthread_exit(
  void *retval
);
```

Arguments:

- *retval if not NULL, copies the exit status into the location pointed to by retval
- **return** status = 0 for success