VI: Interrupts

Section III briefly described the value of interrupts for writing more efficient code. This section teaches how to actually write an interrupt-driven robot. Remember that interrupts are an advanced feature and are not necessary for a beginner to master before trying to write decent robots.

There are three crucial instructions needed by any robot that uses interrupts: SETINT, INTON, and RTI. SETINT tells the processor what label to jump to when the interrupt occurs. INTON turns interrupts on. By default, interrupts are off when a robot first begins combat, so the INTON instruction must appear in the robot's code before interrupts may occur. RTI (ReTurn from Interrupt) is used in place of the RETURN instruction at the end of interrupt-handling routines. RTI reenables interrupts before returning.

Let us begin with a simple example robot, Bozo.

```
# Bozo
# Written 8/22/93 by David Harris
# This robot uses interrupts.

    killem RANGE' SETINT
    INTON

main:
    AIM 5 + AIM' STORE
    main JUMP

killem:
    50 FIRE' STORE
    RTI
```

The first line in the program sets the RANGE interrupt to killem. This causes the robot to automatically call the killem routine when it sights a target (if interrupts are enabled). A complete list of available interrupts appears later in this section. INTON turns on interrupts. The main loop just makes the turret spin.

When a poor target comes into Bozo's sights, i.e. when the RANGE register is non-zero, the RANGE interrupt is triggered. This causes the robot to leave a return address on the stack (just like an IF or CALL statement happened), disable interrupts, and jump to the killem label. The robot will fire a shot with 50 energy. The RTI statement returns from the interrupt by first reenabling interrupts, then jumps back to the return address stored on the top of the stack. RTI is equivalent to an INTON statement followed by a RETURN statement, but is more convenient.

There are three other instructions related to interrupts. One is INTOFF. It disables interrupts until the next INTON instruction. This might be useful in time-critical operations where a robot cannot afford to inadvertently take an interrupt. Another is FLUSHINT, which dumps all pending interrupts from the queue (see below for more information on the interrupt queue). This might be useful if the robot is about to change its course of action and doesn't want the baggage of stale old pending interrupts. The third is SETPARAM. It sets a parameter describing when a particular interrupt should occur. For example, suppose we inserted the following statement right after SETINT:

```
100 RANGE' SETPARAM
```

This sets the parameter on the RANGE interrupt to 100, indicating that a RANGE interrupt should only occur when the RANGE register is less than 100. The various parameters that may be set are also described below. Each interrupt has a default value for its parameter that can be changed if desired by SETPARAM.

If an interrupt is no longer needed, it can be turned off without disabling all other interrupts by setting it to -1, as in the statement:

-1 RANGE' SETINT

All interrupts are initially set to -1 by default.

Summary of Interrupts

The following interrupts are supported in order of highest priority to lowest:

COLLISION
Sets the collision interrupt, to occur whenever the collision register of a robot changes from 0 to 1. SETPARAM has no effect on the collision interrupt.

WALL
Much like collision, but occurs when a robot runs into a wall. SETPARAM also has no effect.

DAMAGE
The damage interrupt is triggered whenever a robot takes damage. SETPARAM sets the minimum threshold required for the damage interrupt to occur; by default it is set to 150. This is useful if a robot should only change its behavior when it is damaged beyond a certain point.

SHIELD
The shield interrupt is triggered whenever a robot's shield transition from above to below a predetermined threshold. SETPARAM sets this threshold; by default it is 25.

TOP
The top interrupt is triggered whenever a robot moves too close to the top wall. SETPARAM determines the y coordinate at which the interrupt is triggered; by default, it is 20. Note that the directional interrupts are only triggered once when the robot crosses the threshold.

BOTTOM (or BOT)
The bottom interrupt is triggered whenever a robot moves too close to the bottom wall. SETPARAM determines the y coordinate at which the interrupt is triggered; by default, it is 280.

LEFT
The left interrupt is triggered whenever a robot moves too close to the left wall. SETPARAM determines the x coordinate at which the interrupt is triggered; by default, it is 20.

RIGHT
The right interrupt is triggered whenever a robot moves too close to the right wall. SETPARAM determines the x coordinate at which the interrupt is triggered; by default, it is 280.

RADAR
The radar interrupt is triggered at the beginning of a chronon or when the aim or scan registers change and the RADAR register is nonzero. SETPARAM determines the maximum distance at which a projectile will set off the interrupt; by default it is 600 to trigger on anything.

RANGE
The range interrupt is much like RADAR, but triggers at either the beginning of a chronon or when the aim or look registers change and RANGE is nonzero. SETPARAM determines the maximum range that will trigger an interrupt and is also 600 by default.

TEAMMATES
The TEAMMATES interrupt is triggered whenever the robot's teammate is killed. The robot can specify

for the interrupt to only occur when the number (excluding yourself) is below a particular value by using SETPARAM. For instance, to only interrupt when all of your teammates are dead, set the parameter to 1. By default, the parameter is set to 5, causing an interrupt when any teammate dies.

ROBOTS
The ROBOTS interrupt is triggered whenever a robot is killed. The robot can specify for the interrupt to only occur when the number (including yourself) is below a particular value by using SETPARAM. For instance, to only interrupt when all other robots are dead, set the parameter to 2. By default, the parameter is set to 6, causing an interrupt when any robot dies.

SIGNAL
The signal interrupt is triggered when data is broadcast over the communication channels by a robot's teammate. SETPARAM determines which channel number is being checked; by default it is channel 0. It is generally a good idea for different robots to transmit on different channels to prevent one robot from overwriting the data sent by the other. Also, since multiple messages that are rapidly sent might be lost, it is generally wise for the RoboTalk hacker to devise some protocol for teammates to acknowledge each other's transmissions.

CHRONON
This interrupt is triggered at the start of each chronon, starting at the chronon set by the parameter. By default, the parameter is set to 0. This interrupt is useful for animated icons or to change behavior after a specific amount of time.

Interrupt Priorities & the Interrupt Queue

Since several different interrupts may occur at the same time, it is important to understand the system RoboWar uses to manage multiple interrupts. The two key concepts are the priority levels of interrupts and the interrupt queue.

Each interrupt has a particular priority. From highest priority to lowest, the interrupts are: COLLISION, WALL, DAMAGE, SHIELD, TOP, BOTTOM, LEFT, RIGHT, RADAR, RANGE, TEAMMATES, ROBOTS, SIGNAL, CHRONON. If two interrupts occur at exactly the same time, the one with higher priority is processed first.

The interrupt queue is a list of interrupts waiting to be processed. If interrupts are disabled (for example, while one interrupt is being handled) and a new interrupt occurs, it may be placed in the queue to be processed when interrupts are reenabled. If there are several interrupts pending in the queue, the one with highest priority is handled first when interrupts become reenabled. Note that RANGE, RADAR, and CHRONON interrupts are never queued (and hence, are not affected by FLUSHINT). Also note that at most one interrupt of each type is stored in the queue; e.g. if a robot is damaged twice while interrupts are disabled, only one damage interrupt will occur when interrupts are reeanbled. Finally, if an interrupt procedure is set to -1 (the default beginning state, or reset explicitly by SETINT), interrupts of that type will not be placed in the queue.

Most interrupts are added to the queue only at the beginnings of chronons. RANGE and RADAR are the exceptions; they can occur midchronon when a robot moves its turret.