# Example unicast in Contiki

```
// the process control block includes variable name for the process and human readable text name

PROCESS(example_unicast_process, "Example unicast");

//the example_unicast_process is to be automatically started when Contiki boots, or, if this module is

compiled as a loadable module, when the module is loaded.

AUTOSTART_PROCESSES(&example_unicast_process);

// Print out an address of sender

static void recv_uc(struct unicast_conn *c, const rimeaddr_t *from)

{

        printf("unicast message received from %d.%d\n",

        from->u8[0], from->u8[1]);

}

// callback function

static const struct unicast_callbacks unicast_callbacks = {recv_uc};

// allocate memory for struct_unicast by defining as static

static struct unicast_conn uc;

PROCESS_THREAD(example_unicast_process, ev, data)

{

// Close unicast connection when a process exits

        PROCESS_EXITHANDLER(unicast_close(&uc);)

 //Define the beginning of a process

        PROCESS_BEGIN();

// sets up a unicast connection with 3 parameters: unicast_struct, uint16_t channel and unicast callback
```

```
        unicast_open(&uc, 146, &unicast_callbacks);
//Start the infinitive loop
        while(1) {
 // declare etimer for an event
                static struct etimer et;
                rimeaddr_t addr;
 // set etimer = 1 second
               etimer_set(&et, CLOCK_SECOND);
 // process waits for an event generated when etimer is expired
               PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));
 // Copy from external data into the packetbuf
                packetbuf_copyfrom("Hello", 5);
               addr.u8[0] = 1;
               addr.u8[1] = 0;
 // Compare two Rime addresses. Return non-zero if the addresses are the same, zero if
//they are different
                if(!rimeaddr_cmp(&addr, &rimeaddr_node_addr)) {
                        unicast_send(&uc, &addr);
                }
        }
        PROCESS_END();
}
```