```
function m6()

    local inst = mc.mcGetInstance('M6 macro') -- This will allow logging the API calls
    with caller information.

        --Start tool change
        mc.mcCntlGcodeExecuteWait(inst, "G90 G53 G0 Z0.0");--Move the Z axis to
        Reference 0.

        locateSpindle, rc = mc.mcSignalGetHandle (inst, mc.OSIG_OUTPUT6) --Turn ON
        spindle lock.
        rc = mc.mcSignalSetState(locateSpindle, 1) --Spindle lock ON
        currentRPM, rc = mc.mcSpindleGetCommandRPM(inst) -- save the current RPM to
        restore later.
        rc = mc.mcSpindleSetCommandRPM(inst, 15) -- use instead of execute M3 S15
        --The possible values to use with the mc.mcSpindleSetDirectionWait() below
        --mc.MC_SPINDLE_OFF
        --mc.MC_SPINDLE_FWD
        --mc.MC_SPINDLE_REV
        rc = mc.mcSpindleSetDirectionWait(inst, mc.MC_SPINDLE_FWD)
        -- There is no input that tells when the keylock has engaged or disengaged.
        wx.wxSleep (5) --Wait 5 seconds for slow rotation to orient spindle in keylock.
        rc = mc.mcSpindleSetDirectionWait(inst, mc.MC_SPINDLE_OFF) --Turn OFF spindle.
        wx.wxSleep (1) --Wait 1 second

        --Carousel pushed OUT to cradle tool oriented in spindle.
        CarouselOut = mc.mcSignalGetHandle (inst, mc.OSIG_OUTPUT5) --Output 5 with 24V
        relay is used to push carousel out.
        mc.mcSignalSetState (CarouselOut, 1) --Carousel OUT motor ON

        -- Use 24V PNP switch (INPUT 5) to stop motor for carousel out.
        local hSig, state
        hSig, rc = mc.mcSignalGetHandle (inst, mc.OSIG_OUTPUT5)
        state, rc = mc.mcSignalGetState(hSig, mc.OSIG_OUTPUT5)
        if (state == 1) then
            rc = mc.mcSignalWait(inst, mc.ISIG_INPUT5, mc.WAIT_MODE_HIGH, 10) -- Wait
            on input 5 to go HIGH 10 seconds.

        -- Always put a timeout in you want to be able to catch errors instead of
        waiting indefinitely.
        -- If you want to wait on the input to go high, use mc.WAIT_MODE_HIGH.  Case is
        important!!!
        -- you really want to check the return code from EVERY API call for
        success/failure.
        -- Below will abort the script gracefully if there is an error.
    end
        if (rc ~= mc.MERROR_NOERROR) then
            wx.wxMessageBox ("Timed out waiting for INPUT 5 to go high.")
            --mc.mcCntlAlarm (inst, 100, "Timed out waiting for input 5 to go high.")
            --return(100) -- 100 here and above is just some number I dreamed up.  you
            could plan your own error number scheme.
        end
        mc.mcSignalSetState(CarouselOut, 0) --Carousel OUT motor OFF
        wx.wxSleep (1) --Wait 1 second

        --Release tool from spindle
        releaseSpindle = mc.mcSignalGetHandle (inst, mc.OSIG_OUTPUT7) --Turn ON air to
        release tool from spindle
        mc.mcSignalSetState (releaseSpindle, 1) --Air ON
        wx.wxSleep (1) --Wait 1 second

        --Move Z axis up 4 inches to clear tool in spindle
        mc.mcCntlGcodeExecuteWait(inst, "G90 G53 G0 Z4.0");--Move the Z axis up 4"
        wx.wxMessageBox ("CHANGE TOOL THEN CLICK OK TO CONTINUE")

        --Move Z axis down 4 inches engage tool
        mc.mcCntlGcodeExecuteWait(inst, "G90 G53 G0 Z0.0");--Move the Z axis down 4" to
        Reference 0.
        wx.wxSleep (1) --Wait 1 second
```

```
58
59          --Engage tool in spindle
60          releaseSpindle = mc.mcSignalGetHandle (inst, mc.OSIG_OUTPUT7) --Turn OFF air to
            grab tool in spindle
61          mc.mcSignalSetState (releaseSpindle, 0) --Air OFF
62          wx.wxSleep (1) --Wait 1 second
63          locateSpindle = mc.mcSignalGetHandle (inst, mc.OSIG_OUTPUT6) --Spindle lock OFF
64          mc.mcSignalSetState (locateSpindle, 0) --Turn OFF spindle lock
65          wx.wxSleep (1)
66
67          --Carousel retracted to clear tool
68          CarouselRetract = mc.mcSignalGetHandle (inst, mc.OSIG_OUTPUT4) --Pull carousel
            IN to disengage tool
69          mc.mcSignalSetState (CarouselRetract, 1) --Carosel IN motor ON
70
71          -- Use 24V PNP switch (INPUT 4) to stop motor for carousel in.
72
73          hSig, rc = mc.mcSignalGetHandle (inst, mc.OSIG_OUTPUT4)
74          state, rc = mc.mcSignalGetState(hSig, mc.OSIG_OUTPUT4)
75          if (state == 1) then
76              rc = mc.mcSignalWait(inst, mc.ISIG_INPUT4, mc.WAIT_MODE_HIGH, 10) -- Wait
                on input 4 to go HIGH 10 seconds.
77          end
78
79          if (rc ~= mc.MERROR_NOERROR) then
80          wx.wxMessageBox ("Timed out waiting for INPUT 4 to go high.")
81          --mc.mcCntlAlarm (inst, 100, "Timed out waiting for input 4 to go high.")
82          --return(100) -- 100 here and above is just some number I dreamed up.  you
            could plan your own error number scheme.
83          end
84          mc.mcSignalSetState(CarouselRetract, 0)
85          --wx.wxSleep (1)
86          end
87
88      if (mc.mcInEditor() == 1) then
89      m6()
90  end
```