```
from machine import Pin, Timer
from time import sleep

# Initialize PIR Motion Sensor
pir = Pin(15, Pin.IN, Pin.PULL_DOWN)
print("PIR Motion Sensor initialized on GP15")

# Initialize Buzzer
buzzer = Pin(16, Pin.OUT)
print("Buzzer initialized on GP16")

# Initialize LCD (Parallel GPIO)
rs = Pin(2, Pin.OUT)  # RS -> GPIO2
e = Pin(3, Pin.OUT)   # E  -> GPIO3
d4 = Pin(4, Pin.OUT)  # D4 -> GPIO4
d5 = Pin(5, Pin.OUT)  # D5 -> GPIO5
d6 = Pin(6, Pin.OUT)  # D6 -> GPIO6
d7 = Pin(7, Pin.OUT)  # D7 -> GPIO7
print("LCD pins initialized")

# LCD dimensions
LCD_LINES = 2
LCD_COLUMNS = 16

# Timer for counting 45 seconds
timer = Timer()

# Flag to check if motion is detected
motion_detected = False

# Countdown value
countdown = 45

# LCD initialization sequence
def lcd_init():
    lcd_command(0x33)  # Initialize
    lcd_command(0x32)  # Set to 4-bit mode
    lcd_command(0x28)  # 2 lines, 5x8 matrix
    lcd_command(0x0C)  # Display on, cursor off
    lcd_command(0x06)  # Increment cursor
    lcd_command(0x01)  # Clear display
    print("LCD initialized")

# Send command to LCD
def lcd_command(cmd):
    rs.value(0)
    write_to_lcd(cmd)
```

```
# Write data to LCD
def lcd_write(data):
    rs.value(1)
    write_to_lcd(data)

# Write 4-bit data to LCD
def write_to_lcd(data):
    # High nibble
    d4.value((data >> 4) & 1)
    d5.value((data >> 5) & 1)
    d6.value((data >> 6) & 1)
    d7.value((data >> 7) & 1)
    pulse_enable()
    # Low nibble
    d4.value(data & 1)
    d5.value((data >> 1) & 1)
    d6.value((data >> 2) & 1)
    d7.value((data >> 3) & 1)
    pulse_enable()

# Pulse the enable pin
def pulse_enable():
    e.value(1)
    sleep(0.0005)
    e.value(0)
    sleep(0.0005)

# Clear LCD
def lcd_clear():
    lcd_command(0x01)
    print("LCD cleared")

# Print string to LCD
def lcd_print(message, line):
    lcd_command(0x80 | (0x40 * line))  # Set cursor to line
    for char in message:
        lcd_write(ord(char))
    print(f"LCD printed: {message} on line {line}")

# Update countdown on LCD
def update_countdown():
    global countdown
    lcd_print(f"Time left: {countdown:2d}s", 1)  # Display countdown on line 1
    if countdown > 0:
        countdown -= 1

# Timer callback
def on_timer(timer):
    global motion_detected, countdown
    if motion_detected:
        if countdown > 0:
            update_countdown()  # Update countdown on LCD
        else:
            print("Timer elapsed: Buzzer ON, LCD message displayed")
            buzzer.value(1)  # Turn on buzzer
            lcd_clear()
            lcd_print("Stop wasting water!", 0)
            lcd_print("Skibidi toilet!", 1)
            sleep(5)  # Display message for 5 seconds
            buzzer.value(0)  # Turn off buzzer
            lcd_clear()
            motion_detected = False
            countdown = 45  # Reset countdown
            print("Buzzer OFF, LCD cleared")

# PIR callback
def pir_callback(pin):
    global motion_detected, countdown
    if pir.value() == 1:  # Motion detected
        if not motion_detected:
            motion_detected = True
            countdown = 45  # Reset countdown
            print("Motion detected: Starting 45-second timer")
            timer.init(mode=Timer.PERIODIC, period=1000, callback=on_timer)  # 1-second
intervals

# Initialize LCD
lcd_init()
lcd_clear()

# Attach interrupt to PIR sensor
pir.irq(trigger=Pin.IRQ_RISING, handler=pir_callback)
print("PIR interrupt attached")

# Main loop
print("Starting main loop")
while True:
    sleep(0.1)
    lcd_print("here", line)
```