Web access for VisionSOM

October 2020

SoMLabs

SOM
System on Module

CB
Carrier Board

DK
Development Kit

Engineering
Since 2003 delivering proven designs

Agenda

- Adding web packages to image
- Network connection through USB
- Python Flask application examples

Exercises

- /home/dev/Excercises/Workshop4/
- Lab1 - Configuring the Ethernet Gadget
- Lab2 - Hello world for Python Flask
- Lab3 - Dynamic HTML templates
- Lab4 - Controlling on-board LEDs through web browser
- Lab5 - Monitoring on-board buttons through web browser

Adding web packages

Adding web packages

- meta-somlabs/recipes-fsl/images/fsl-image-validation-imx.bbappend

```
PACKAGE_INSTALL += " \
    somlabs-demo \
    dhcp-server \
    apache2 \
    python-flask \
    "
```

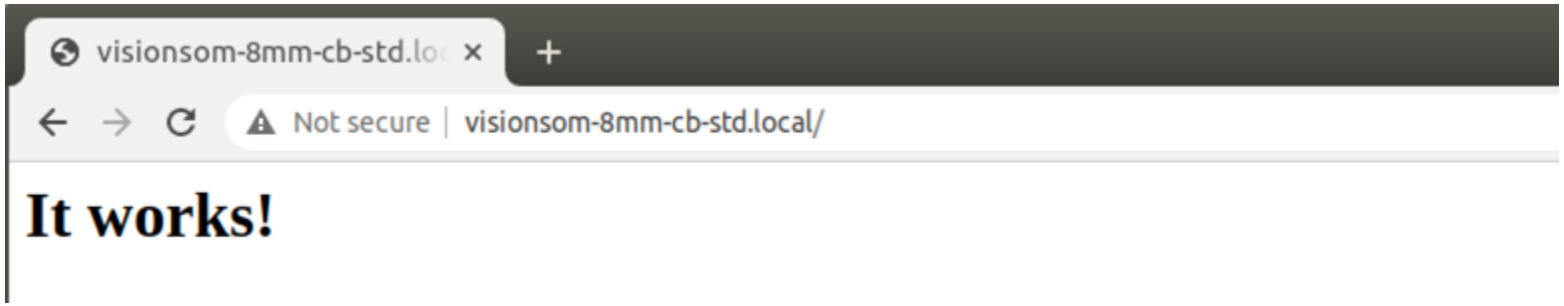
Adding web packages

- build-visionsom-8mm-cb-std/conf/bblayers.conf

```
BBLAYERS += "${BSPDIR}/sources/meta-somlabs"
BBLAYERS += "${BSPDIR}/sources/meta-openembedded/meta-webserver"
```

Adding web packages

- Verify web access
  - visionsom-8mm-cb-std.local
  - <IP Address>



- /usr/share/apache2/default-site/htdocs/index.html

USB Ethernet Gadget

USB Ethernet Gadget

- Source files (Excercises/Workshop4/Lab1):
  - dhcpd.conf
  - usb-bringup
  - usb-bringup.service

USB Ethernet Gadget

- Configuring the DHCP server (/etc/dhcp/dhcpd.conf)

```
subnet 192.168.7.0 netmask 255.255.255.0 {
  range 192.168.7.2 192.168.7.100;
}
```

USB Ethernet Gadget

- Bringing up the usb0 interface (/sbin/usb-bringup)

```
#!/bin/sh

modprobe g_ether
sleep 1
ip addr add 192.168.7.1/24 dev usb0
ip link set usb0 up
sleep 1
systemctl start dhcpd
```

- Set executable flag

```
chmod u+x usb-bringup
```

USB Ethernet Gadget

- Startup service (/etc/systemd/system/usb-bringup.service)

[Unit]

Description=USB0 interface bringup service

After=systemd-networkd.service

[Service]

ExecStart=/sbin/usb-bringup

Type=oneshot

[Install]

WantedBy=multi-user.target

- Enable the service

```
systemctl enable usb-bringup.service
```

Python Flask web application

Python Flask web application

- Excercises/Workshop4/Lab2

```
mkdir /usr/share/somlabs-web
scp main.py root@visionsom-8mm-cb-std.local:/usr/share/somlabs-web
```

- Application structure (/usr/share/somlabs-web/main.py)

```
from flask import Flask
app = Flask(__name__)

@app.route('/')
def main():
      return 'Hello VisionSOM!'
```

Python Flask web application

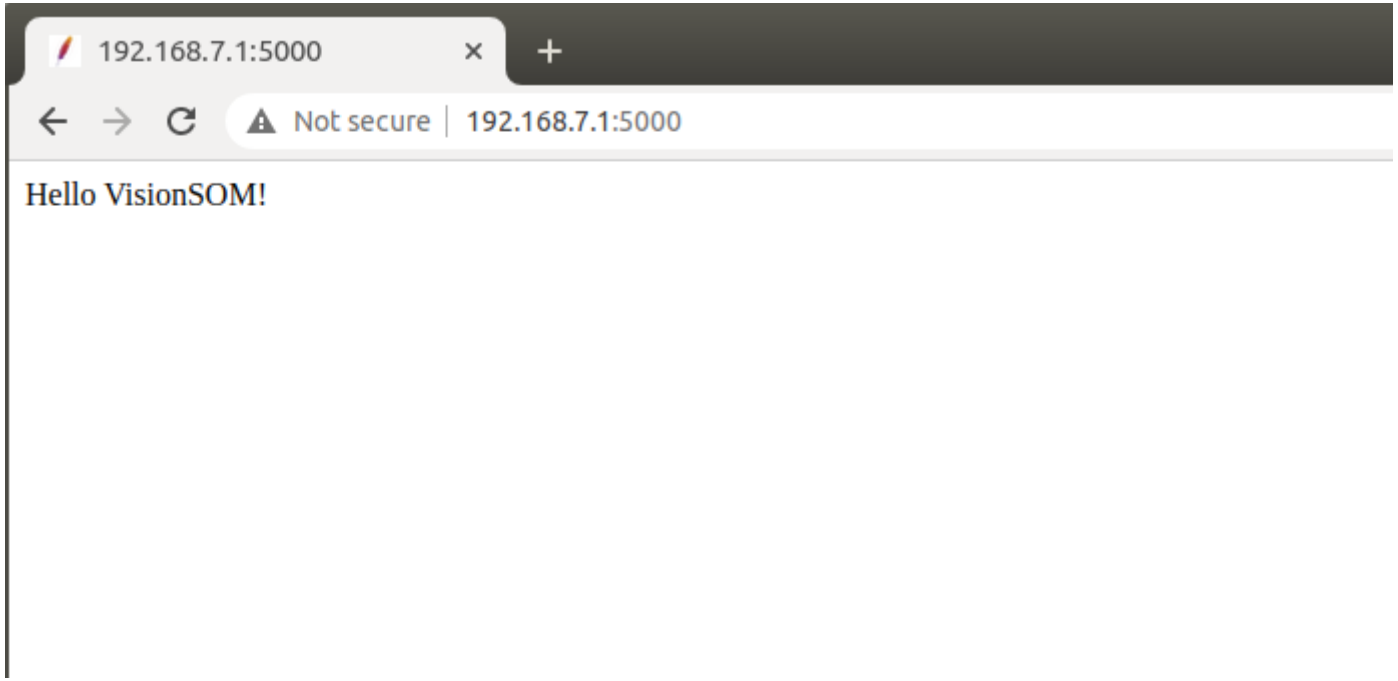
- Running the application

FLASK_APP=main.py flask run --host=0.0.0.0 --port=5000

```
 * Serving Flask app "main.py"
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
192.168.7.6 - - [01/Oct/2020 08:36:32] "GET / HTTP/1.1" 200 -
```

Python Flask web application

- Web access (port 5000)



Python Flask - dynamic HTML templates

Python Flask - HTML templates

- Excercises/Workshop4/Lab3
  - main.py
  - static/style.css
  - templates/main.html

```
scp -r somlabs-web root@visionsom-8mm-cb-std.local:/usr/share/
```

Python Flask - HTML templates

- main.py file

```
from flask import Flask
from flask import send_from_directory
from flask import render_template
from datetime import datetime

app = Flask(__name__, template_folder='/usr/share/somlabs-web')
```

Python Flask - HTML templates

- main.py file

```
@app.route('/favicon.ico')
def favicon():
    return send_from_directory('/usr/share/apache2/default-
site/htdocs/manual/images/',
                               'favicon.ico',
mimetype='image/vnd.microsoft.icon')


@app.route('/')
def main():
    return render_template('templates/main.html', date=str(datetime.now()))
```

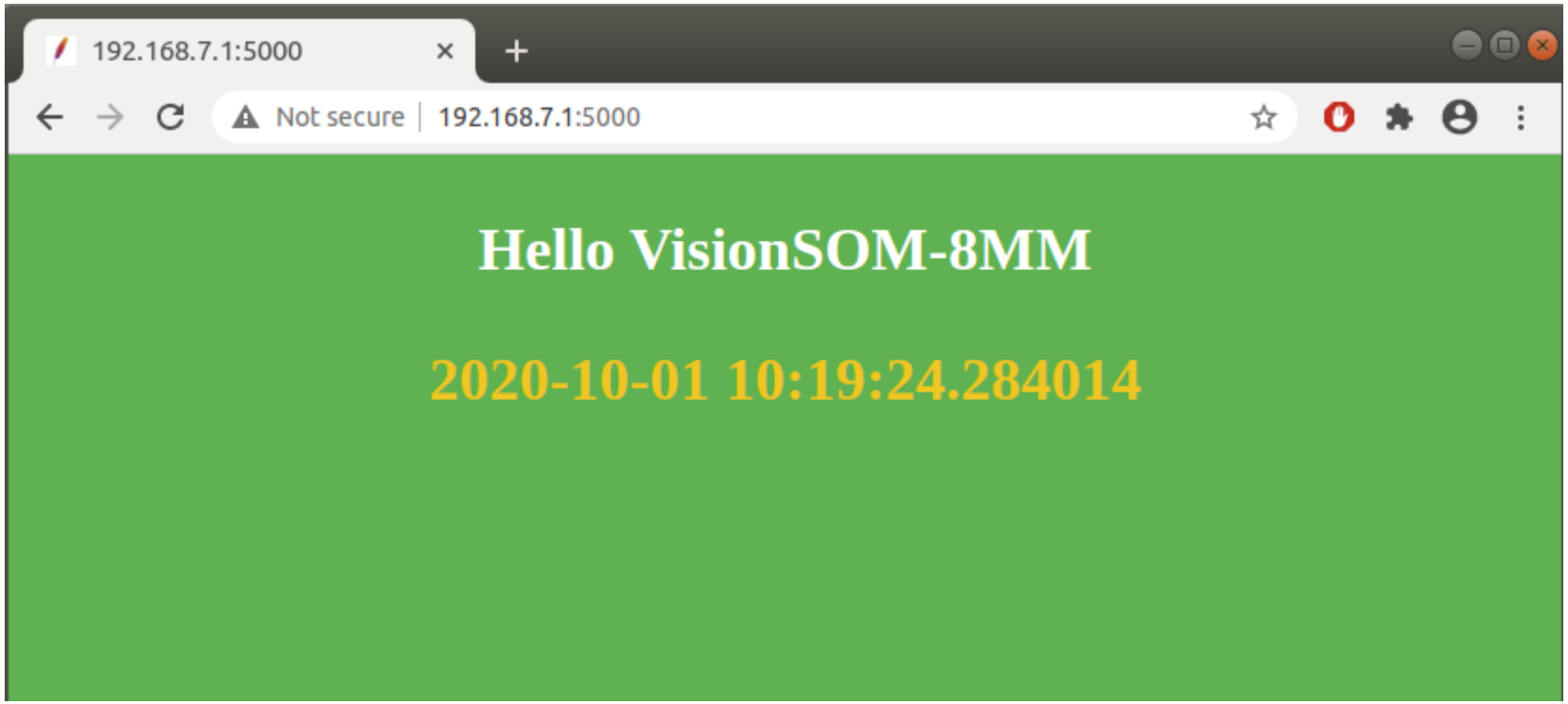
Python Flask - HTML templates

- HTML template

```
<!DOCTYPE html>
<html>
  <head>
    <link rel= "stylesheet" type= "text/css" href= "{{ url_for('static',
filename='style.css') }}">
  </head>
  <body>
    <p class="header white">Hello VisionSOM-8MM</p>
    <p class="header yellow">{{ date }}</p>
  </body>
</html>
```

Python Flask - HTML templates

- cd /usr/share/somlabs-web/
- FLASK_APP=main.py flask run --host=0.0.0.0 --port=5000



Python Flask - controlling on-board LEDs

Python Flask - controlling on-board LEDs

- Excercises/Workshop4/Lab4
  - leds.py
  - static/style.css
  - templates/leds.html

```
scp -r somlabs-web root@visionsom-8mm-cb-std.local:/usr/share/
```

Python Flask - controlling on-board LEDs

- leds.py file

```
@app.route('/', methods=['GET', 'POST'])
def leds():
  if request.method == "POST":
    set_led('/sys/class/leds/LED-IO-04/brightness', request.form.get('led1')=='on')

  led1_state = get_led('/sys/class/leds/LED-IO-04/brightness')
  return render_template('templates/leds.html',
                         led1='checked' if led1_state=='1' else '')
```

Python Flask - controlling on-board LEDs

- leds.py file

```
@def get_led(led_path):
    f = open(led_path, 'r')
    state = f.read()[0]
    f.close()
    return state

def set_led(led_path, enabled):
    f = open(led_path, 'w')
    f.write('{}'.format('1' if enabled else '0'))
    f.close()
```

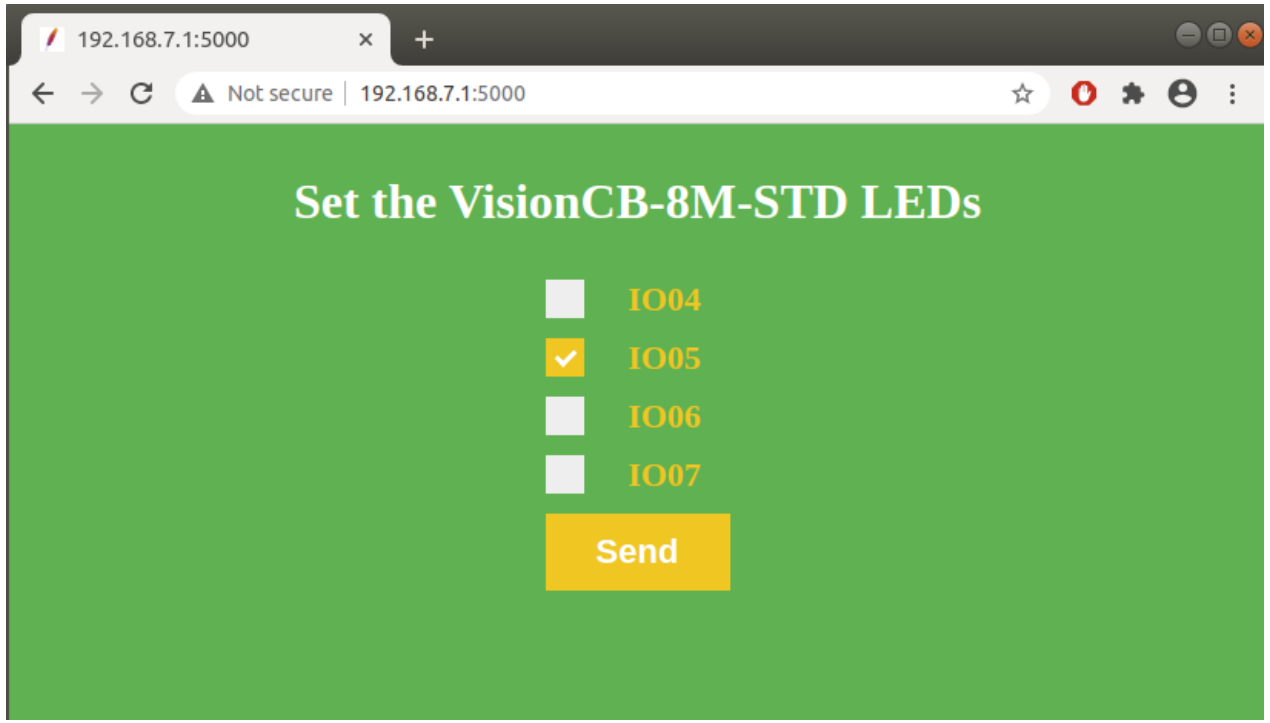
Python Flask - controlling on-board LEDs

- HTML template

```
<body>
  <p class="header white">Set the VisionCB-8M-STD LEDs</p>

  <form method=post>
    <label class="container">IO04
      <input type="checkbox" name=led1 {{led1}}>
      <span class="checkmark"></span>
    </label>

    <input type=submit value=Send>

  </form>
</body>
</html>
```

Python Flask - controlling on-board LEDs

- cd /usr/share/somlabs-web/
- FLASK_APP=leds.py flask run --host=0.0.0.0 --port=5000



Python Flask - monitoring on-board buttons

Python Flask - controlling on-board buttons

- Excercises/Workshop4/Lab5
  - buttons.py
  - static/style.css
  - templates/buttons.html

```
scp -r somlabs-web root@visionsom-8mm-cb-std.local:/usr/share/
```

Python Flask - controlling on-board buttons

- buttons.py file

```
@app.route('/stream')
def stream():
    def eventStream():
        while True:
            yield 'data: {}\n\n'.format(get_message())
    return Response(eventStream(), mimetype="text/event-stream")

@app.route('/')
def buttons():
    return render_template('templates/buttons.html')
```

Python Flask - controlling on-board buttons

- buttons.py file

```
def get_message():
    f = open( "/dev/input/event0", "rb" );
    while 1:
        event_data = struct.unpack('4IHHI', f.read(24))
        if event_data[4]==1:
            f.close()
            if event_data[6]==0:
                return ""
            if event_data[5]==259:
                return "IO3"
```

Python Flask - controlling on-board buttons

- HTML template

```
<body>
  <p class="header white">Active button on VisionCB-8M-STD</p>
  <p id="target_container" class="header yellow"></p>
</body>
<script type="text/javascript">
  var targetContainer = document.getElementById("target_container");
  var eventSource = new EventSource("/stream")
  eventSource.onmessage = function(e) {
    targetContainer.innerHTML = e.data;
  };
</script>
</html>
```

Python Flask - controlling on-board buttons

- cd /usr/share/somlabs-web/
- FLASK_APP=buttons.py flask run --host=0.0.0.0 --port=5000

- Connecting the Future
- Customer Centric