INTRODUCING CYBER LABS INTO ENGINEERING COURSES AND DEVELOPING CURRICULUM LEADING TO A SPECIALIZATION

KALYAN MONDAL, PI, NSA GRANT

ANGELA ELIAS-MEDINA, RESEARCH ASSISTANT, NSA GRANT

PRESENTATION FOR THE CAE SYMPOSIUM, MIAMI, FL

THURSDAY, NOVEMBER 8, 2018

PROJECT SPONSORED BY THE NATIONAL SECURITY AGENCY UNDER GRANT/COOPERATIVE AGREEMENT ENTITLED 'CYBERSECURITY WORKFORCE EDUCATION - CNAP INITIATIVES' NUMBER H98230-17-1-0321

ANY OPINIONS, FINDINGS, AND CONCLUSIONS OR RECOMMENDATIONS EXPRESSED IN THIS MATERIAL ARE THOSE OF THE AUTHOR(S) AND DO NOT NECESSARILY REFLECT THE VIEWS OF THE NATIONAL SECURITY AGENCY

HIGHLIGHTS OF THE NSA GRANT PROPOSAL

1.2.1. *INTEGRATION OF HANDS-ON LEARNING EXPERIENCES INTO CYBERSECURITY CURRICULUM.*

- DEVELOP HANDS-ON LABS & CURRICULUM THAT SUPPORT
  - SECURE EMBEDDED SYSTEMS DESIGN
    - EXISTING COURSES: GRADUATE EMBEDDED SYSTEMS, UNDERGRADUATE MICROPROCESSOR SYSTEM DESIGN II
  - HARDENING OPERATING SYSTEM AND SYSTEMS PROGRAMMING
    - EXISTING COURSES: GRADUATE OPERATING SYSTEMS, GRADUATE SYSTEMS PROGRAMMING
  - INTRODUCING DATA ANALYTICS TO CYBERSECURITY
    - EXISTING COURSE: GRADUATE/UNDERGRADUATE CYBERSECURITY
- STUDY FEASIBILITY OF DEVELOPING ENGINEERING AND/OR COMPUTING SPECIALIZATIONS

PRINCIPAL INVESTIGATORS & THEIR ROLES

- **DR. KALYAN MONDAL**, ASSOCIATE PROFESSOR OF ELECTRICAL ENGINEERING, PI – PROJECT MANAGEMENT, MICROCONTROLLER LAB AND CURRICULUM DEVELOPMENT
- **DR. WILLIAM PHILLIPS**, LECTURER OF COMPUTER SCIENCE, CO-PI – CYBERSECURITY SECURITY ENGINEERING COURSE AND LAB DEVELOPMENT
- **DR. RAVI RAO**, ASSISTANT PROFESSOR OF ELECTRICAL ENGINEERING, CO-PI – SECURE EMBEDDED SYSTEMS COURSE AND LAB DEVELOPMENT
- **DR. ALEX RUDNIY**, CONSULTANT, CO-PI – BIG DATA TOOLS AND LAB DEVELOPMENT FOR CYBERSECURITY

GRADUATE EMBEDDED SYSTEMS COURSE

Embedded Sys Fa 16

Programming in C

Embedded Systems Theory

Keil MCB2130 Board
7 labs

Procure Raspberry-PI & Setup Lab

Embedded Sys Fa 18

Programming in Python and C

Linux & Networking

Embedded Systems Theory & Security Concepts

Raspberry-PI based
6 NEW labs

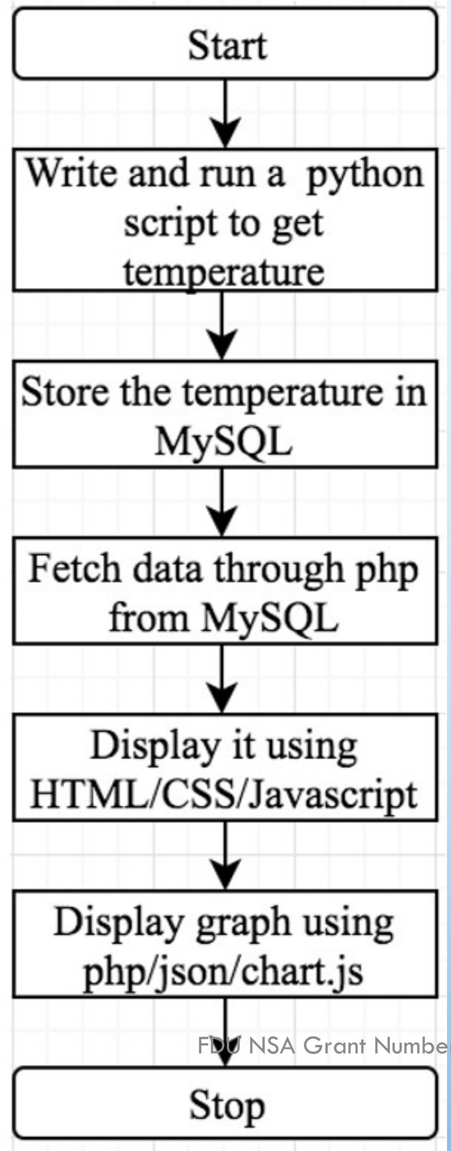
INTRODUCING CYBER CONCEPTS INTO EMBEDDED SYSTEMS COURSE

- USE THE RASPBERRY PI TO PROVIDE HANDS-ON LEARNING
- DESIRABLE FEATURES INCLUDE - LOW COST, MULTIPLE I/O PORTS, GPIO PINS
- ALLOWS EASY BREAD-BOARDING
- ALSO IDEALLY SUITED FOR CYBERSECURITY TRAINING AS VULNERABLE SOFTWARE CAN BE INSTALLED UNDER SAFE SANDBOX

EMBEDDED SYSTEMS COURSE UPDATE

- THE COURSE PREVIOUSLY TAUGHT THEORETICAL CONCEPTS
  - SENSORY DATA ACQUISITION & PROCESSING
  - PROCESSOR ARCHITECTURE
  - CONCURRENT PROGRAMMING
  - INTER-PROCESS COMMUNICATION
  - COMBINED WITH A LAB USING THE ARM PROCESSOR
- ADDITIONAL CONTENT WAS INTRODUCED AS FOLLOWS
  - 4 WEEK INTRODUCTION TO PYTHON USING ZYBOOKS ONLINE PLATFORM
  - 2 WEEK INTRODUCTION TO BASIC LINUX OPERATIONS
  - 1 WEEK FOR BASIC USAGE OF RASPBERRY PI WITH NETWORK CONNECTIVITY
  - 1 WEEK TO UNDERSTAND BASIC NETWORKING AND FIREWALLS (IP TABLES)
  - 1 WEEK TO UNDERSTAND A SIMPLE APPLICATION: CONTINUOUS TEMPERATURE SENSING

Second Lab: Temperature Sensing & Processing



Start

Write and run a python script to get temperature

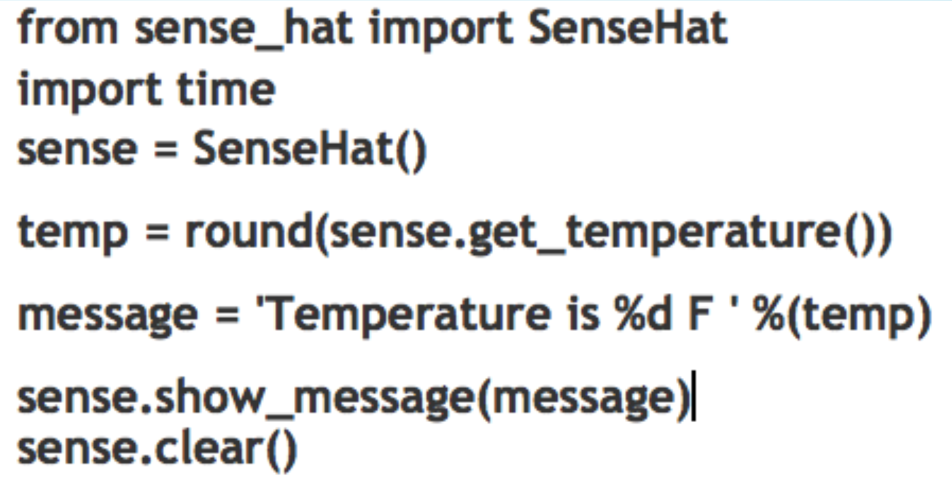
Store the temperature in MySQL

Fetch data through php from MySQL

Display it using HTML/CSS/Javascript

Display graph using php/json/chart.js

Stop



```
from sense_hat import SenseHat
import time
sense = SenseHat()
temp = round(sense.get_temperature())
message = 'Temperature is %d F ' %(temp)
sense.show_message(message)
sense.clear()
```

Details:

Rao et al, IEEE STEM Education Conference, Princeton NJ, 2018

UG MICROCONTROLLER BASED DESIGN & SECURITY

Microcontroller Course Sp 18

Programming in C

Microcontroller Based Design Theory

Dragon 12-Plus Board
10 labs

Procure Equipment & Setup Lab

Microcontroller Course Sp 20

Programming in Python and C

Security Concepts

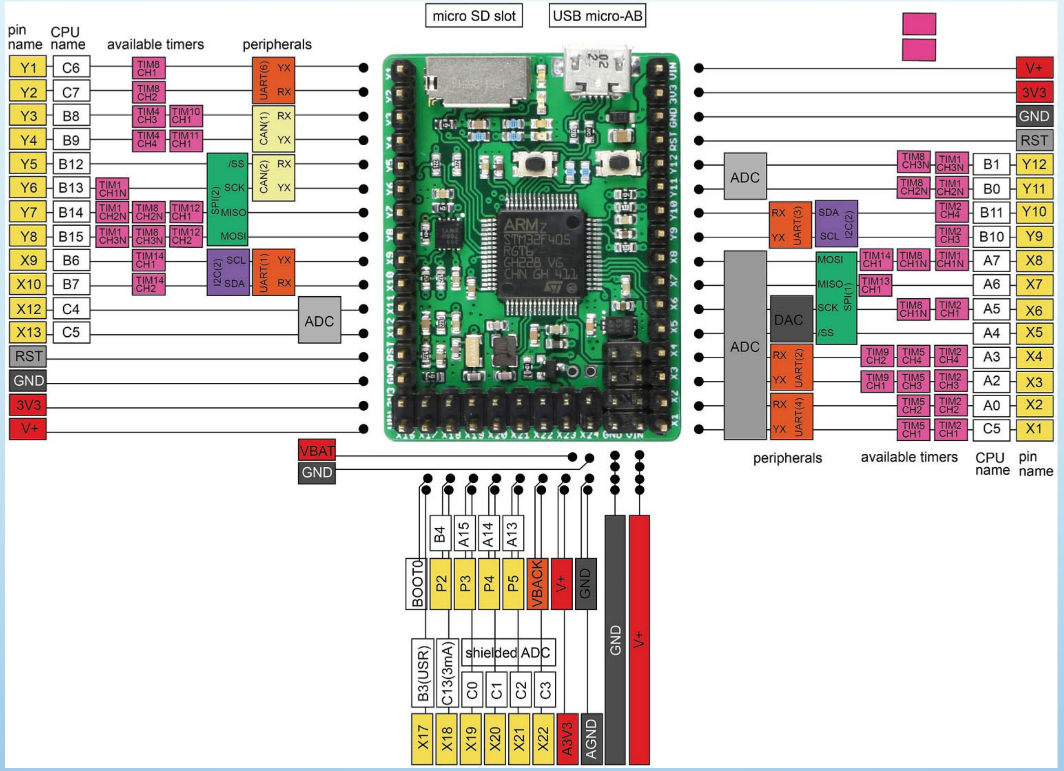
Microcontroller Based Design Theory

NEW labs

MICROPYTHON & PYBOARD FOR UG MICRO COURSE

MicroPython

- lean implementation of Python 3.4 with a subset of Python library
- optimized to run on microcontrollers


micro SD slot
USB micro-AB
pin name
CPU name
available timers
peripherals
Y1 C6 TIM8 CH1 UART(6) YX
Y2 C7 TIM8 CH2 RX
Y3 B8 TIM4 CH3 TIM10 CH1 CAN(1) RX
Y4 B9 TIM4 CH4 TIM11 CH1 YX
Y5 B12 SPI(2) /SS CAN(2) RX
Y6 B13 TIM1 CH1N SCK YX
Y7 B14 TIM1 CH2N TIM8 CH2N TIM12 CH1 MISO
Y8 B15 TIM1 CH3N TIM8 CH3N TIM12 CH2 MOSI
X9 B6 TIM14 CH1 I2C(2) SCL UART(1) YX
X10 B7 TIM14 CH2 SDA RX
X12 C4 ADC
X13 C5
RST
GND
3V3
V+
VBAT
GND
V+
3V3
GND
RST
Y12 B1 TIM1 CH3N TIM8 CH3N ADC
Y11 B0 TIM1 CH2N TIM8 CH2N
Y10 B11 TIM2 CH4 I2C(2) SDA UART(3) RX
Y9 B10 TIM2 CH3 SCL YX
X8 A7 TIM1 CH1N TIM8 CH1N TIM14 CH1 SPI(1) MOSI ADC
X7 A6 TIM13 CH1 MISO
X6 A5 TIM2 CH1 TIM8 CH1N SCK DAC
X5 A4 /SS
X4 A3 TIM2 CH4 TIM5 CH4 TIM9 CH2 UART(2) RX
X3 A2 TIM2 CH3 TIM5 CH3 TIM9 CH1 YX
X2 A0 TIM2 CH2 TIM5 CH2 UART(4) RX
X1 C5 TIM2 CH1 TIM5 CH1 YX
peripherals
available timers
CPU name
pin name
BOOT0 B3(USR) X17
P2 B4 C13(3mA) X18
P3 A15 C0 X19
P4 A14 C1 X20
P5 A13 C2 X21
VBACK C3 X22
shielded ADC
V+ A3V3
GND AGND
GND
V+

Pyboard

- compact circuit board
- runs MicroPython
- it connects to PC over USB
- USB flash drive to save Python scripts
- inexpensive
- pin level programming
- Python prompt (a REPL) for instant programming
- 16 k Data RAM
- 256 k code space

UG MICROCONTROLLER COURSE

- MICROCONTROLLER ARCHITECTURE
- INTRODUCE DRAGON-12 PLUS BOARD & ASSOCIATED CODEWARRIOR IDE
- C BASED PORT PROGRAMMING
- REAL-TIME INTERRUPT BASED PORT PROGRAMMING
- SIGNAL GENERATION BY MASKABLE INTERRUPT
- TIMER BASED SIGNAL MEASUREMENT
- TIMER BASED WAVEFORM GENERATION
- ADC & TEMPERATURE SENSING
- 12 HANDS-ON LABS ON DRAGON-12 PLUS BOARD

FUTURE ENHANCEMENTS TO THE MICROCONTROLLER COURSE

- INTRODUCE MICROPYTHON
- INTRODUCE PYBOARD
- INTRODUCE MICROPYTHON/PYBOARD BASED LABS
- INTRODUCE BASIC ENCRYPTION/DECRYPTION CONCEPTS
- ONE OR MORE LABS EMPHASIZING DATA SECURITY

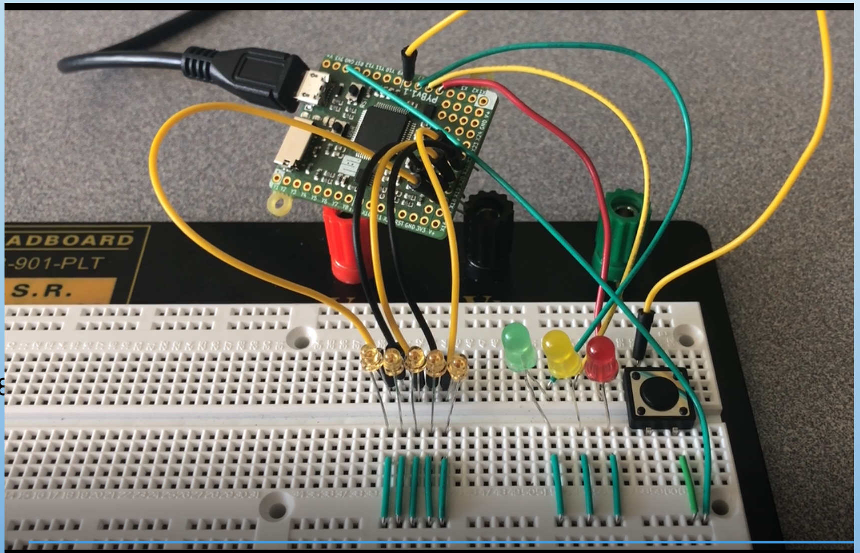
UG NEW MICROCONTROLLER LABS

- LAB 1 – MOSTLY INTRODUCES PIN LEVEL PROGRAMMING USING MICROPYTHON ON PYBOARD
    - PIN CONNECTED SEVEN SEGMENT DISPLAY OF PATTERNS
    - NO SECURITY CONCERNS
- LAB 2 – TIMER CYCLE BASED TRAFFIC LIGHT CONTROLLER
    - SECURITY CONCERNS ONLY IF WIRELESS/NETWORK CONTROL OF LIGHTS
- LAB 3 – ENCRYPTING TEMPERATURE/PRESSURE SENSOR DATA FOR PORTING & PROCESSING
    - DATA ENCRYPTION DRIVES SECURITY CONCERNS & MITIGATION

LAB 2: TIMER BASED TRAFFIC LIGHT CONTROLLER

- CYCLE THROUGH LED LIGHTING:
  - RED -> RED -> RED -> RED -> GREEN -> GREEN -> GREEN -> GREEN -> YELLOW
- NEED TO IMPORT PYB AND MICROPYTHON LIBRARIES
- DECLARE A CLASS “SEMAFORO(OBJECT)” WITH THE METHODS:
  - INITIALIZATION “_INIT_” FOR TIMER, LED AND ANY OTHER VARIABLES
  - INTERRUPT SERVICE ROUTINE “ SIGNAL” PASSING TIMER PARAMETERS
- INITIALIZE TIMER OBJECT WITHIN “_INIT_”
  - TIM = PYB.TIMER(2,PRESCALER=10,PERIOD=10000000)   # DEFAULT SINGLE PERIOD
  - TIMER 2 TRIGGERS INTERRUPT AT FREQUENCY = 84 MHZ /10 / 10000001 ~ 0.84 HZ
- INITIALIZE CYCLECOUNT & LED COUNT WITHIN “_INIT_”
- PERFORM TIMER CALLBACK OF INTERRUPT SERVICE ROUTINE “SIGNAL” WITHIN “_INIT_”
  - TIM.CALLBACK(SELF.SIGNAL)

```
# Timer Based Traffic Light Controller
# Assuming both RED & GREEN LEDs stay ON 4 times as long as YELLOW does
#
import pyb, micropython
micropython.alloc_emergency_exception_buf(100)

class Semaforo(object):

    def __init__(self):
        self.led=[1,2,3,1]            # R,G,Y,R
        self.count = 0                # Initialize led count
        tim = pyb.Timer(2,prescaler=10,period=10000000)
                                      # default single period
        self.cyclecount = 0           # Initialize cycle count
        tim.callback(self.signal)
```



LAB 2: CYCLE-BASED CODING

- Define ISR “SIGNAL”
  - Increment CYCLECOUNT
  - Check if CYCLECOUNT = 4, turn GREEN LED ON
  - Check if CYCLECOUNT = 8, turn YELLOW LED ON
  - Check if CYCLECOUNT = 9, turn RED LED ON
    - Reset all counters to restart the cycle
- Start the program with RED LED ON
- Long hand code, can be optimized

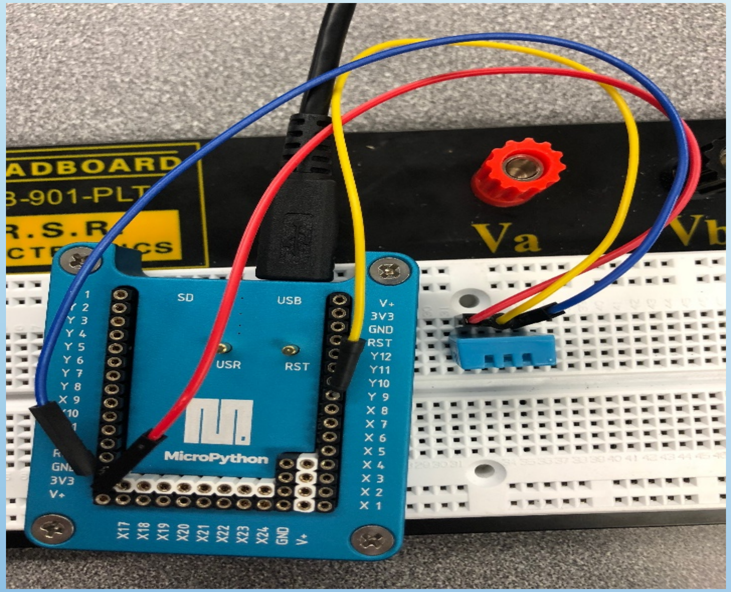
Student lab exercises may involve varying default timer period, varying durations for signal lights, additional three lights for the orthogonal direction, etc.

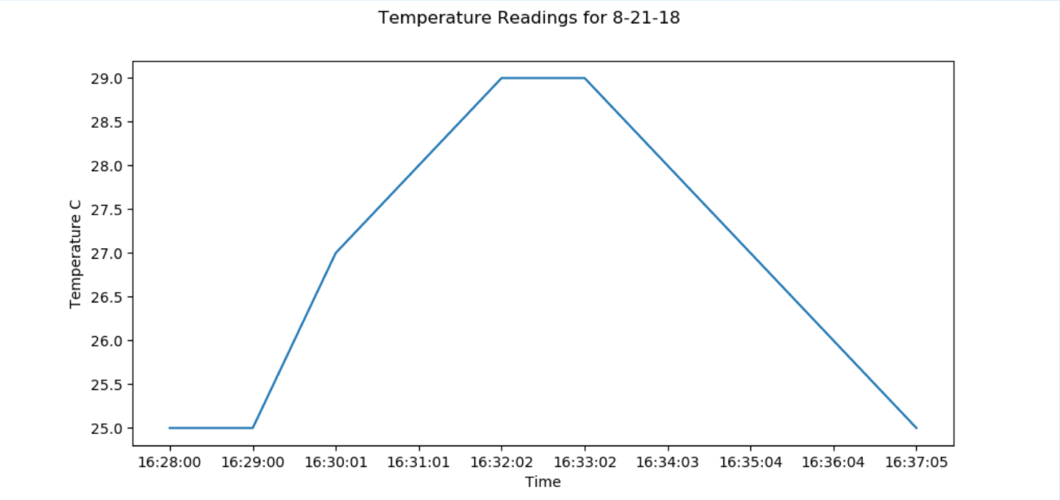
No external control - Security hazards are minimal!

```
  def signal(self, tim):
      self.cyclecount += 1                # increment cycle count

      if self.cyclecount == 4:
          pyb.LED(self.led[self.count]).toggle()        # turns RED off
          pyb.LED(self.led[self.count+1]).toggle()      # turns GREEN on
          self.count += 1
      elif self.cyclecount == 8:
          pyb.LED(self.led[self.count]).toggle()        # turns GREEN off
          pyb.LED(self.led[self.count+1]).toggle()      # turns YELLOW on
          self.count += 1
      elif self.cyclecount == 9:
          pyb.LED(self.led[self.count]).toggle()        # turns YELLOW off
          pyb.LED(self.led[self.count+1]).toggle()      # turns RED on
          self.count = 0                                # reset led count
          self.cyclecount = 0                           # reset cycle count

#---------#
pyb.LED(1).toggle()                     #program starts with Red on

Semaforo()                              #interrupt class is called
```

LAB 3: ENCRYPTION & DECRYPTION OF SENSOR DATA

- Real time acquisition of DHT-11 sensor data in CSV format
- Encrypting and saving in a file
- Porting to a PC
- Decrypt & display





LAB 3: DATA ACQUISITION & ENCRYPTION

```
# Get values for date and time.
print('Enter numerical values for date and time.')
#year,month=int(input('Year: ')), int(input('Month: '))
day,wday=int(input('Day: ')), int(input('Weekday (1-7;Mon-Sun): '))
hour,minute=int(input('Hour (24hr-day): ')), int(input('Minute: '))
second=int(input('Second: '))

# Set up RTC with given values
rtc = pyb.RTC()
rtc.datetime((2018,8,day,wday,hour,minute,second,0))

delay = 30000 # milliseconds
filename = 'data_log.csv'

try:
    print('\nCtrl+C to stop.')
    dataLog(delay, filename)
    while True:
        file_length = len(open(filename).readlines())
        if file_length < 10:
            dataLog(delay, filename)
        else:
            read_write(delay, filename)

except:
    print('Program stopped.')
```

LAB 3: DECRYPT & DISPLAY

```
# main()
dt_array = [] # array for decrypted datetime
T_array = []  # array for decrypted temperature
log = open('data_log.csv','r')
enc_str = log.readline().split(',')

while enc_str:
    try:
        dec_dt = decrypt(enc_str[0])
        dec_T = int(decrypt(enc_str[1]))
        dt_array.append(dec_dt[11:])
        T_array.append(dec_T)
        print(dec_dt, dec_T)
        key_ok = 1
        enc_str = log.readline().split(',')
    except:     # EOF
        break
log.close()

# Show plot if correct key is given
try:
    if key_ok==1:
        plt.plot(dt_array,T_array)
        plt.suptitle('Temperature Readings for 8-23-18')
        plt.xlabel('Time')
        plt.ylabel('Temperature C')
        plt.show()
except:
    print('Incorrect Key')
```

LABS IN GR. ADVANCED SYSTEMS PROGRAMMING

Systems Programming Sp 17

Programming in C: Linux & Windows

Operating System Theory + 6 Lab Projects

Systems Programming Theory + 6 UNIX API Labs

Procure Beaglebone Black & Setup Lab

Programming in C: Linux & Windows

Operating System Theory + UNIX programming projects

Systems Programming Theory + 6 UNIX API Labs

NEW Beaglebone Black based 6 embedded labs

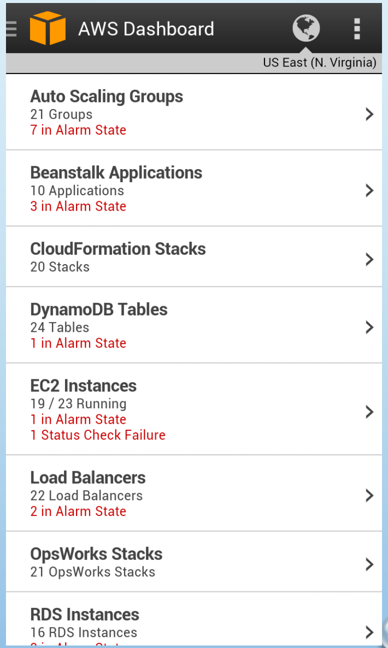
DATA ANALYTICS BASED LABS IN CYBERSECURITY

Cybersecurity Sp 17

Foundations of Cybersecurity

13 Open Source SW based labs

Procure Server/AWS plus Datasets & Setup Apache Metron



AWS Dashboard

US East (N. Virginia)

Auto Scaling Groups
21 Groups
7 in Alarm State

Beanstalk Applications
10 Applications
3 in Alarm State

CloudFormation Stacks
20 Stacks

DynamoDB Tables
24 Tables
1 in Alarm State

EC2 Instances
19 / 23 Running
1 in Alarm State
1 Status Check Failure

Load Balancers
22 Load Balancers
2 in Alarm State

OpsWorks Stacks
21 OpsWorks Stacks

RDS Instances
16 RDS Instances

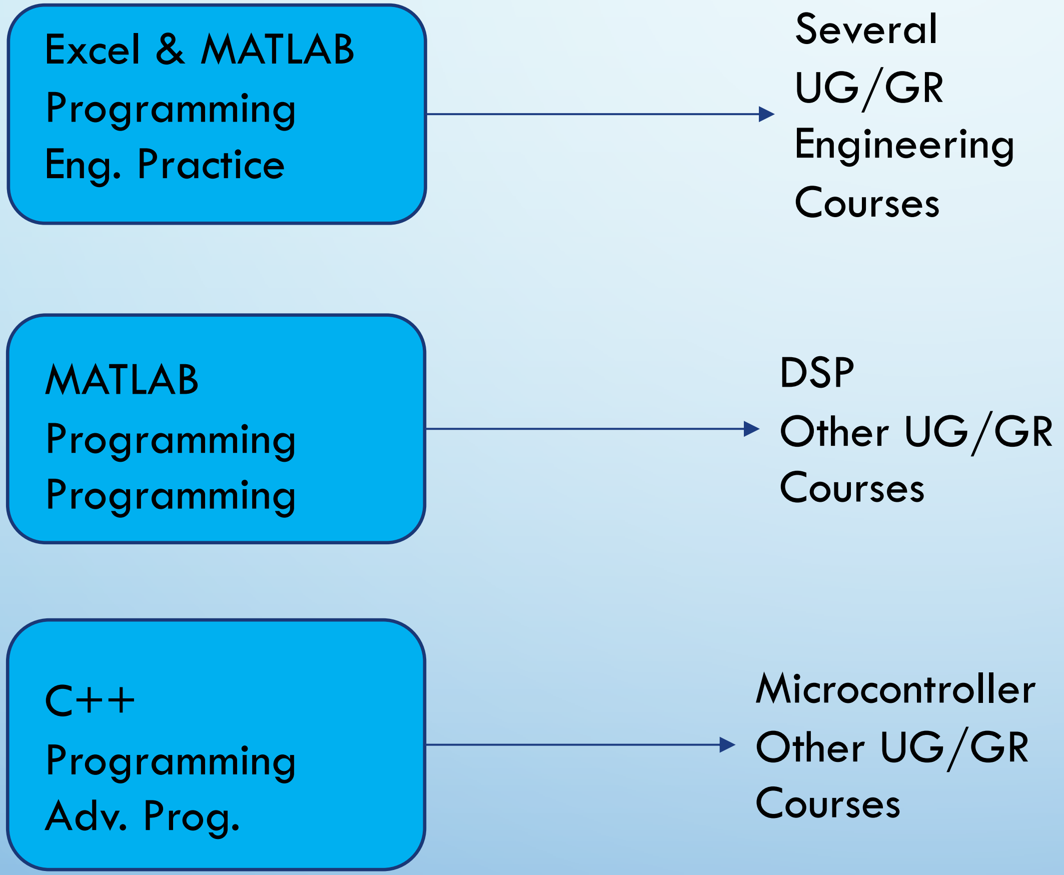
Foundations of Cybersecurity & Data Analytics

3 NEW Security Labs

UG CURRICULUM DEVELOPMENT - PYTHON

- IEEE SPECTRUM RANKED PYTHON #1 TWO YEARS IN A ROW
- DISCUSSIONS ON INTRODUCING PYTHON IN CS/IT IAB
  - POSITIVE ENDORSEMENT
  - TWO SEMESTERS OF JAVA CAN BE REPLACED BY PYTHON & JAVA
- DISCUSSIONS ON INTRODUCING PYTHON IN EE/EET IAB
  - MIXED FEEDBACK
  - CAN INTRODUCE PYTHON W/O DISPLACING ANY LANGUAGE TAUGHT NOW

PROGRAMMING LANGUAGES TAUGHT IN UG ENGINEERING



Excel & MATLAB
Programming
Eng. Practice

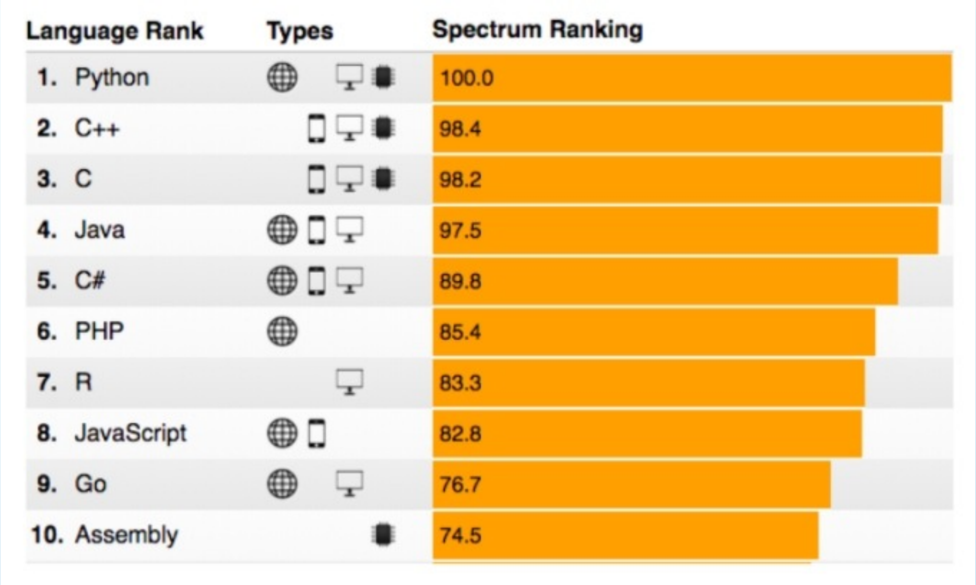
Several
UG/GR
Engineering
Courses

MATLAB
Programming
Programming

DSP
Other UG/GR
Courses

C++
Programming
Adv. Prog.

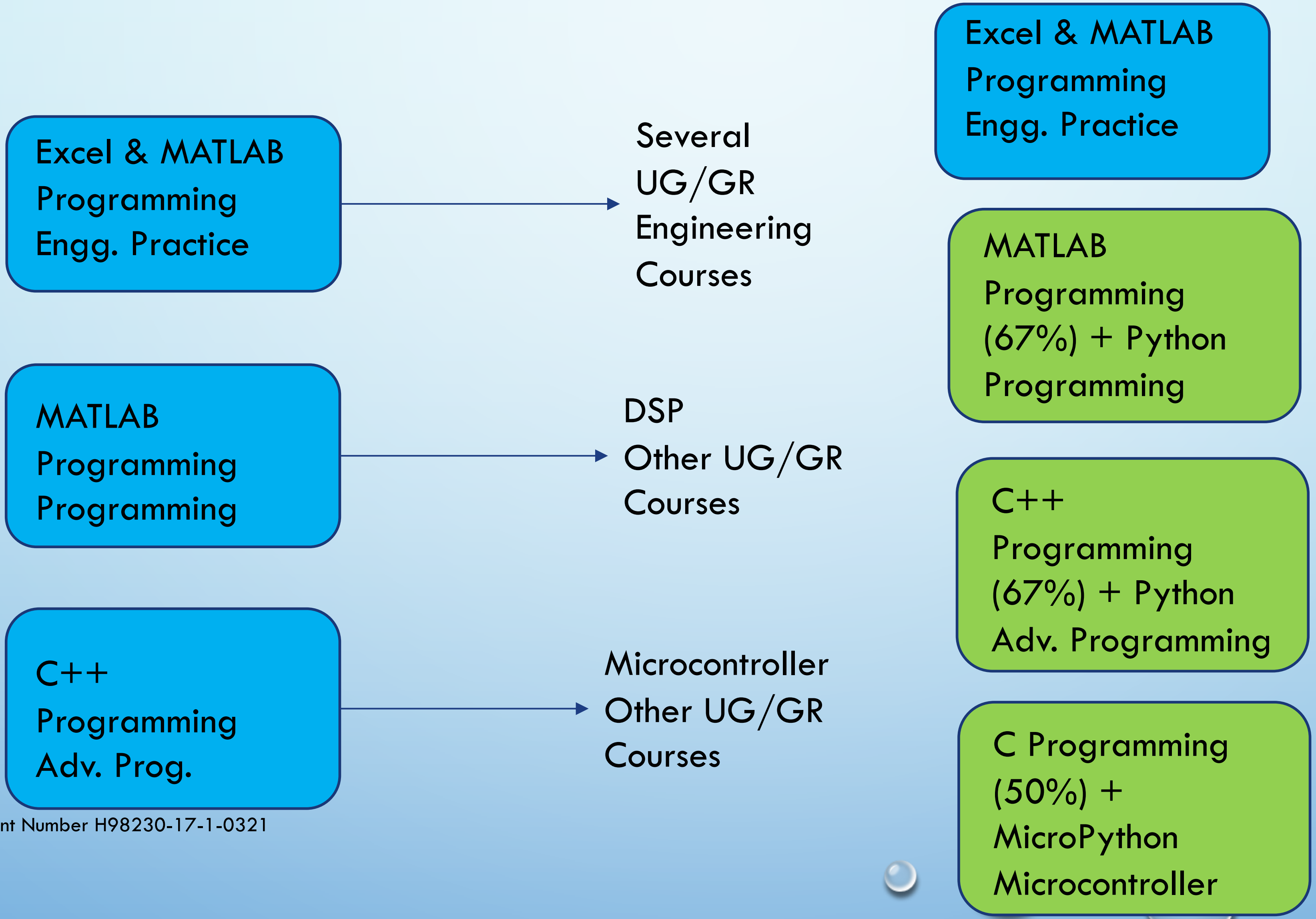
Microcontroller
Other UG/GR
Courses



Language Rank	Types	Spectrum Ranking
1. Python	🌐 🖥 ■	100.0
2. C++	📱 🖥 ■	98.4
3. C	📱 🖥 ■	98.2
4. Java	🌐 📱 🖥	97.5
5. C#	🌐 📱 🖥	89.8
6. PHP	🌐	85.4
7. R	🖥	83.3
8. JavaScript	🌐 📱	82.8
9. Go	🌐 🖥	76.7
10. Assembly	■	74.5

Introduce Python

INTRODUCE PYTHON IN UG ENGINEERING



Excel & MATLAB Programming Engg. Practice

Several UG/GR Engineering Courses

Excel & MATLAB Programming Engg. Practice

MATLAB Programming Programming

DSP Other UG/GR Courses

MATLAB Programming (67%) + Python Programming

C++ Programming Adv. Prog.

Microcontroller Other UG/GR Courses

C++ Programming (67%) + Python Adv. Programming

C Programming (50%) + MicroPython Microcontroller

GRADUATE COURSEWARE ENHANCEMENTS

- SPECIALIZATION IN SECURE EMBEDDED SYSTEMS (MSEE, MSCPE)
  - MEET EMBEDDED SYSTEMS (EBS) KU REQUIREMENTS
  - ENHANCE EXISTING COURSES BASED ON KU MAPPING
  - INTRODUCE 1 OR 2 NEW COURSES
- MEET SYSTEMS PROGRAMMING (SPG) KU REQUIREMENTS (MSCS)
  - ENHANCE EXISTING COURSES BASED ON KU MAPPING
- SPECIALIZATION IN DATA SECURITY ANALYSIS (HELP DEVELOP MS DATA SCIENCE)
  - ENHANCE EXISTING COURSES BASED ON KU MAPPING
  - INTRODUCE NEW COURSES AS NEEDED

GR SECURE EMBEDDED SYSTEMS SPECIALIZATION

NSA KU	FDU Course
Basic Cryptography	Cybersecurity
Network Defense	Cybersecurity
Basic Networking	Computer Networks
Basic Scripting & Programming	Intro to Computer Programming
Operating Systems Concepts	Operating Systems
Cyber Threats	Information Security
Policy, Legal, Ethics, and Compliance	Information Security

NSA KU	FDU Course
Embedded Systems	Embedded Systems
Hardware/Firmware Security	*To develop a new course based on research*
Life-cycle Security	Information Security
Low Level Programming	Micro. System Design
Network Technology & Protocols	Computer Communication Networks
QA/Functional Testing	*To enhance Software Engineering*
Secure Programming Practices	*To develop Secure Software Development*

GR DATA SECURITY ANALYSIS SPECIALIZATION

NSA KU	FDU Course
Basic Scripting and Programming	Intro to Computer Programming
Basic Networking	Computer Networks
Security Program Management	Information Security
Security Risk Analysis	Cybersecurity
Data Administration	Big Data Analytics with Hadoop and R

NSA KU	FDU Course
Databases	Database Systems
IA Architectures	Cybersecurity
IA Compliance	Information Security
Intrusion Detection/Prevention Systems	Firewalls and Intrusion-Detection Systems
Systems Security Engineering	*To develop Secure SW Development*
Wireless Sensor Networks	*New course outline developed*

CONCLUSIONS

- GREAT SUCCESS IN OBTAINING EXTRAMURAL FUNDING
  - FDU'S CAE-CDE DESIGNATION
  - FACULTY STRENGTHS & RESOURCES
  - MULTI-DISCIPLINARY APPROACH INVOLVING EE, CS, CPE
  - STRONG SUPPORT BY THE GRANTS OFFICE
- MAJOR LEARNING/RESEARCH OPPORTUNITIES FOR FACULTY & STUDENTS
  - SEVERAL RESEARCH STUDENTS DID WELL IN THE JOB MARKET
  - FACULTY BETTER ATTUNED TO NSA/DHS CAE-CDE REQUIREMENTS

CONCLUSIONS

- **MULTIPLE CONFERENCE PRESENTATIONS**
  - **CAE TECH TALK, PRINCETON IEEE STEM CONFERENCE, CISSE, CAE SYMPOSIUM**
- **ENHANCED EMBEDDED SYSTEMS COURSE WITH RASPBERRY-PI LABS**
- **INTRODUCED NEW COURSE CYBERSECURITY SYSTEMS ENGINEERING WITH BEAGLEBONE BLACK LABS**
- **INTRODUCED NEW DATA ANALYTICS BASED LABS OVER AMAZON WEB SERVICES IN CYBERSECURITY COURSE**
- **CAN INTRODUCE PYTHON IN UG EE/CS CURRICULUM IN NEAR FUTURE**
- **WILL INTRODUCE PYBOARD BASED LABS IN UG MICROCONTROLLER BASED DESIGN COURSE**
- **CAN START TWO NEW CAE-CDE SPECIALIZATIONS IN NEAR FUTURE**