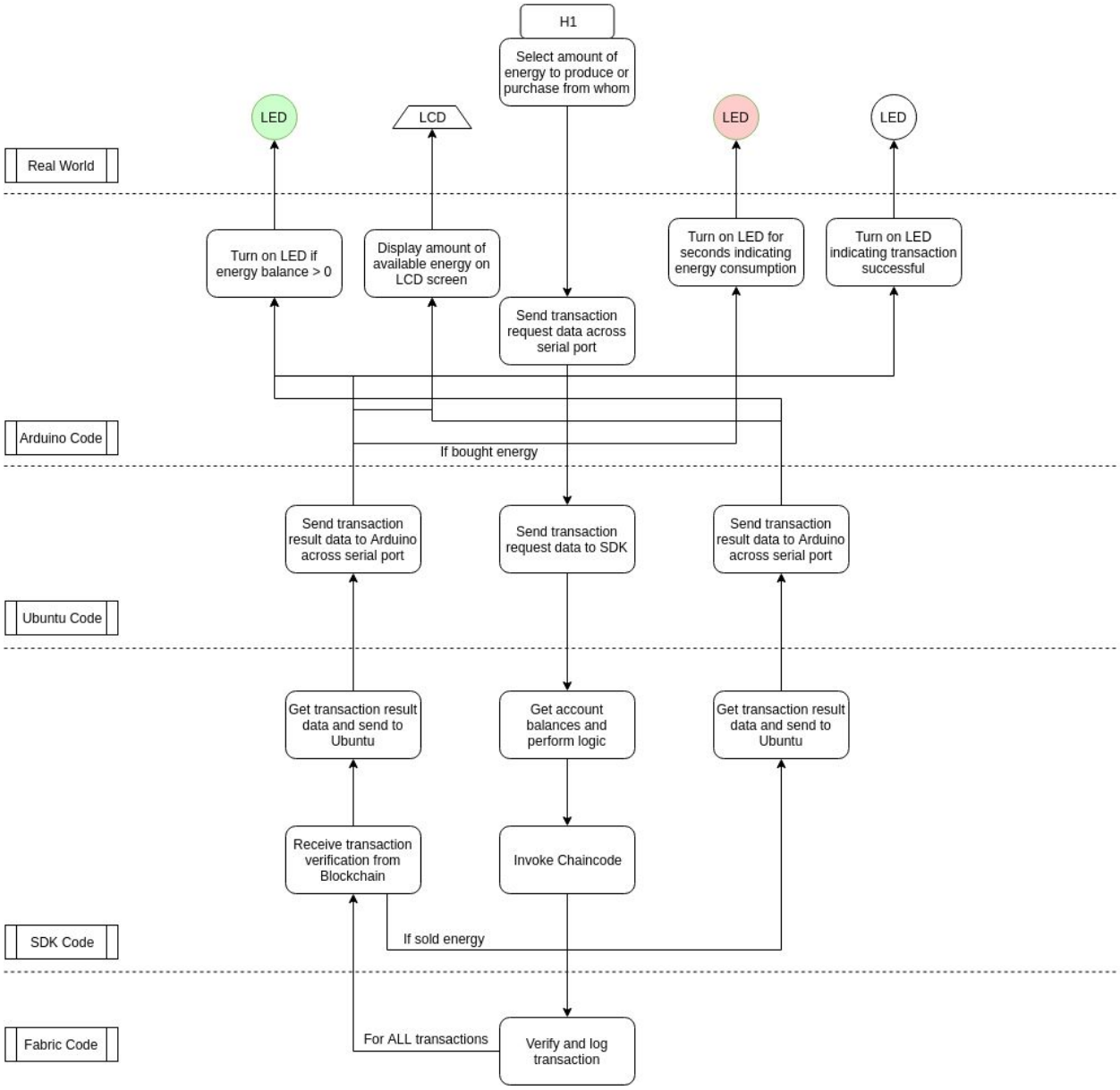
LOW LEVEL OVERVIEW



Mechanical Devices & Arduino 101

Digital I/O (3.3VDC)

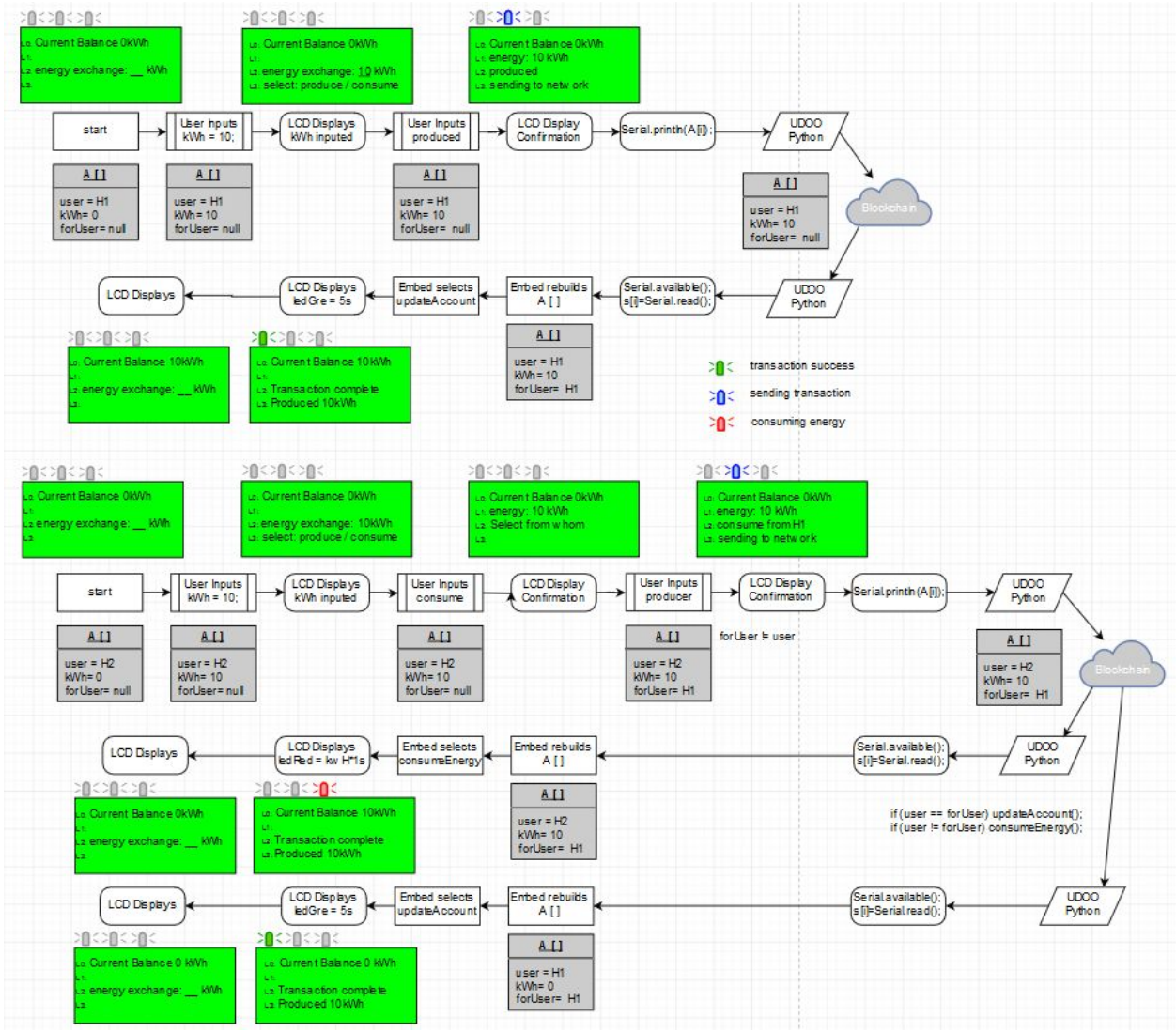
Utilizes Arduino IDE with code written in C

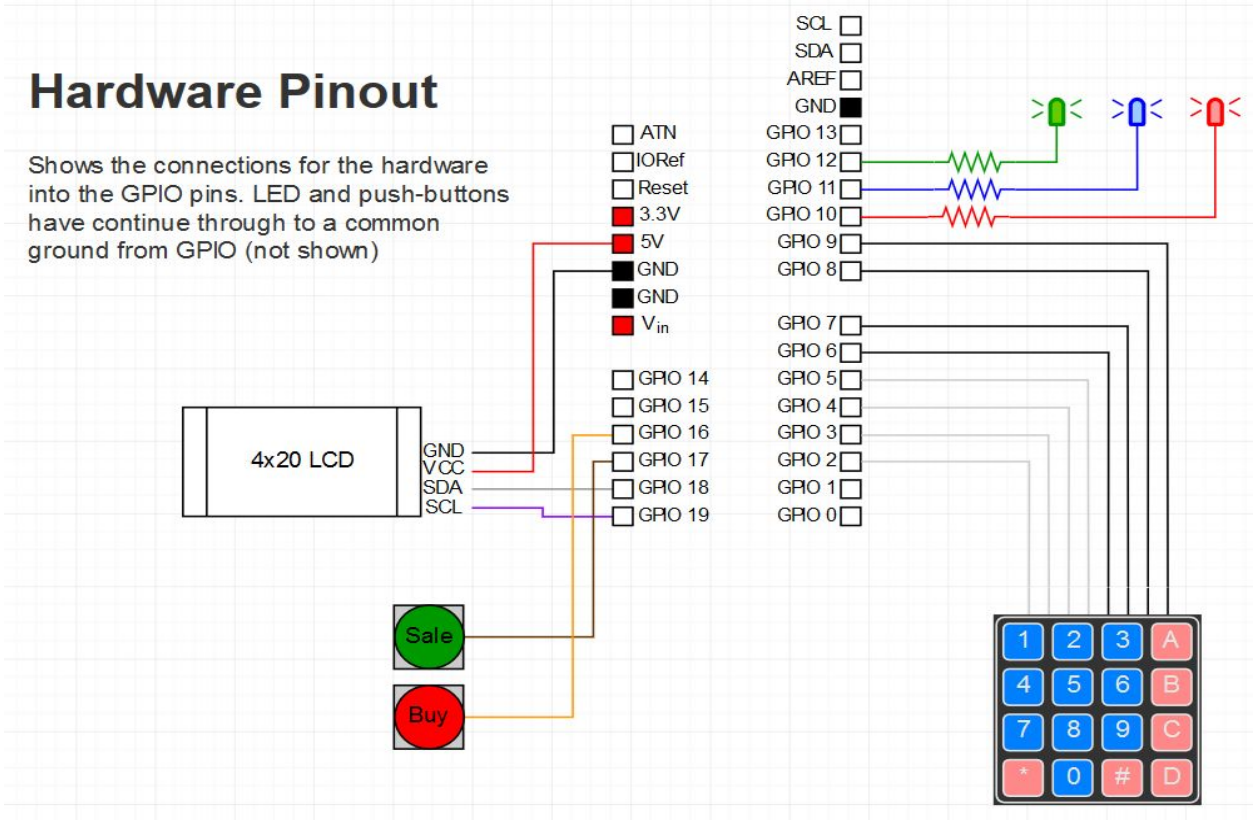
Can produce and/or purchase up to 99kW of energy (2 digits)

Allows for data to be deleted and re-entered

Sends transaction request data to serial port as a string

Receives transaction result and account balance data from serial port as a string





Arduino & UDOO

Code written in Javascript utilizing nodejs
UDOO monitors serial port /dev/ttyACM0 for transaction request data
Receives string delimited by \n
Sends transaction result including account balance and energy consumption to serial port as a string

Transaction Request Array Format:
Arduino -> [a, b, c] -> UDOO
a = Requestor
b = Energy Amount
c = Seller

If “c” = 0, the request is a Produce function where “a” (Requestor) is producing power internally.
If “c” != 0, the request is a Consume function where “a” (Requestor) is purchasing energy from “c” (Seller).

Transaction Result Array Format:
UDOO -> [x, y, z] -> Arduino
x = Target
y = Energy Consumed or Account Balance
z = Seller

If “z” = “0”, then “y” is the Account balance of “x” (Target).
If “z” != “0”, then “y” is Amount of energy consumed by “x” (Target) from “z” (Seller).
If “x” = “z”, then “x” is the Amount of energy consumed by “x” by itself “x”.

Arduino nodes ignore any Transaction Result array that does not match their house number in the 1st element. Example: Transaction Result [3 10 0] is ignored by all Arduino nodes except House 3.

For every transaction, the Arduino will receive 2 Transaction Result arrays. For a Produce Function where House 1 produces 15 kWh, the following arrays are sent to all Arduinos:
UDOO -> [1 15 0] -> Arduino
UDOO -> [0 0 0] -> Arduino

The 0 array is ignored by every Arduino node and is a filler used for only Consume Functions.
For a Consume Function where House 1 consumes all 25 kWh from House 3's balance, the following arrays are sent to all Arduinos:
UDOO -> [1 25 3] -> Arduino
UDOO -> [3 0 3] -> Arduino

The 1st array is the consume result for House 1 to consume for 25 seconds. The 2nd array is to update House 3's account balance to 0 kWh.

Current code for sending arduino transactionRequest to SDK and pushing transactionResult to Arduino can be viewed at https://github.com/austinfifield/udoo.git

UDOO & SDK App

Code written in Javascript utilizing nodejs
Triggers event when new transactionRequest data has been received
Exports transactionRequest string via websocket to localhost:3000
Gets transaction result data to include account balances from SDK websocket on localhost:3000
All transactionResult strings should be broadcast to ALL UDOO localhost websockets for each transaction result

UDOO will posts transactionRequest data and export the module as an array:
For Produce Function:

```
[1 10 0]  // House 1 produces 10kWh of energy
```

For Consume Function:

```
[1 15 2]  // House 1 buys 15 kWh from House 2
```

SDK will post transactionResult data and export the module as an array:
For Produce Function:

```
[1 10 1]  // House 1 account balance is 10kWh
[0 0 0] // Filler array with no data. Ignored by Arduino
```

For Consume Function:

```
[1 15 2]  // House 1 consumes 15kWh from House 2
[2 0 2]  // House 2 account balance is 0kWh
```

SDK App & Hyperledger (UNDER CONSTRUCTION - STEVE 3/1/19 7:00 pm)
Setup - Prerequisites (Ubuntu 16.04 or 18.04)
CHAINCODE SETUP
Code snippets to be pasted into the terminal will be color coded CYAN

Following the instructions per Hyperledger Fabric Docs (steps listed below link) :
https://hyperledger.github.io/composer/latest/installing/installing-prereqs#ubuntu

If cURL is not installed:

```
Sudo apt-get update && sudo apt-get install curl
```

Hyperledger Fabric prerequisites

```
curl -O https://hyperledger.github.io/composer/latest/prereqs-ubuntu.sh
chmod u+x prereqs-ubuntu.sh
./prereqs-ubuntu.sh
```

LOG OUT AND THEN LOG BACK IN
Install Golang

```
sudo add-apt-repository ppa:gophers/archive
sudo apt-get update
sudo apt-get install golang-go
```

Make go directory

```
mkdir ~/go && cd ~/go && mkdir src && cd src
mkdir github.com && cd github.com
mkdir deblk && mkdir hyperledger && cd hyperledger
```

Hyperledger Fabric v1.1 (For Huy's network)

```
git clone -b release-1.1 https://github.com/hyperledger/fabric.git
```

Install chaincode

```
cd ../deblk/
git clone https://github.com/theDweeb/temp-cc
```

Open bash.bashrc to set environment variable path

```
sudo gedit /etc/bash.bashrc
```

If gedit is not installed

```
sudo apt-get install gedit
```

PASTE THE EXPORTS BELOW AT THE END OF BASH.BASHRC

```
export CHANNEL=foo
export GOPATH=$HOME/go
export FABRIC_CC_SRC=$GOPATH/src/github.com/deblk/temp-cc
export CRYPTO_CONFIG=/$HOME/Fabric/fabric-network/networkup/docker/crypto/v1.1/crypto-config*
export HLFBIN1_1=/$HOME/Fabric/fabric-network/networkup/docker/binaries/v1.1
export FABRIC_VERSION=hlfv11
alias dockerps='docker ps --format '\"table {{.Names}}\t{{.Image}}\t{{.Status}}\t{{.Ports}}\"'
```

Install Huy's network

```
mkdir ~/Fabric && cd ~/Fabric
git clone https://github.com/httran13/fabric-network.git
cd ~/Fabric/fabric-network/networkup/docker
./fabric.sh upAll
```

Verify everything is working correctly

```
dockerps
```

You should see this:

```
steve@dweeb:~/Fabric/fabric-network/networkup/docker$ dockerps
NAMES                                   IMAGE
dev-peer0.org1.example.com-defaultcc-v1 dev-peer0.org1.example.com-defaultcc-v1-
cli                                     hyperledger/fabric-tools:x86_64-1.1.0
peer0.org1.example.com                  hyperledger/fabric-peer:x86_64-1.1.0
orderer.example.com                     hyperledger/fabric-orderer:x86_64-1.1.0
capeerorg1                              hyperledger/fabric-ca:x86_64-1.1.0
couchdb0                                hyperledger/fabric-couchdb
>5984/tcp
configtxlator                           hyperledger/fabric-tools:x86_64-1.1.0
```

Along with status and ports. The main thing to make sure is the dev-peer0.org1 is running and not peer0.org1. If you get peer0.org1 instead of dev-peer0.org1 then do the following:

```
./fabric.sh down && ./fabric.sh clean
./fabric.sh upAll
```

MAKE SURE YOU GET THIS WORKING BEFORE MOVING ON

Install Huy's SDK (for fresh network)

```
cd ~/Fabric
git clone https://github.com/httran13/fab-node-express.git
```

Or install the latest version used for senior design

```
git clone https://github.com/theDweeb/fab-node-express
```

Install node version manager (nvm) follow their instructions on the github page below
https://github.com/creationix/nvm

MAKE SURE YOU USE NODEJS VERSION 8.9.0

Paste this in terminal to lock 8.9.0.

```
nvm install 8.9.0
nvm alias default 8.9.0
node --version
```

MAKE SURE YOUR CONTAINERS ARE RUNNING

```
dockerps or docker ps
```

SDK SETUP (MAKE SURE YOUR NODE VERSION IS 8.9.0 AND DOCKER CONTAINERS ARE RUNNING)

```
cd ~/Fabric/fab-node-express
npm install
```

Create a file inside fab-node-express and name it ".env". Open this file and paste the following inside:

```
ENVIRONMENT = 'local'
FABRIC_LOCAL_CP = 'cp-local.json'
```

This will set the environment variables needed to run the SDK

Run the test to verify everything is working

```
npm run test
```

You should see all three tests passing.

IF YOU SEE THIS ERROR

```
steve@dweeb:~/Fabric/fab-node-express$ npm run test

> fab-client@0.0.0 test /home/steve/Fabric/fab-node-express
> ENVIRONMENT=local mocha --timeout 10000 ./tests/integration/*.js



(node:19925) DeprecationWarning: grpc.load: Use the @grpc/proto-loader module with grpc.
  Testing end2end with network
[03-05-2019 21:42:11] [fabric/fabric-interface.js] [DEBUG]:     Creating user :testS5kpU
[03-05-2019 21:42:11] [fabric/user-utils.js] [DEBUG]:    Created new User obj:{"name":"te
[03-05-2019 21:42:11] [fabric/user-utils.js] [ERROR]:    Failed to create user!
[03-05-2019 21:42:11] [fabric/user-utils.js] [ERROR]:    Calling enrollment endpoint fail
[03-05-2019 21:42:11] [fabric/fabric-interface.js] [ERROR]:     Error creating user:Erro
(node:19925) UnhandledPromiseRejectionWarning: Unhandled promise rejection (rejection id
(node:19925) [DEP0018] DeprecationWarning: Unhandled promise rejections are deprecated.
```

Delete the hfc-cvs and hfc-kvs folders and run the test again.

IF YOU SEE THIS ERROR

```
[03-05-2019 21:44:57] [fabric/fab-utils.js] [DEBUG]:     Sending off proposalResponses, ALL GOOT
[03-05-2019 21:44:57] [fabric/fab-utils.js] [DEBUG]:     Successfully sent Proposal and received Propos
[03-05-2019 21:44:57] [fabric/fab-utils.js] [DEBUG]:     Successfully send transaction to order
[03-05-2019 21:44:57] [fabric/fabric-interface.js] [INFO]:      Invoke successful with txId: 7f914f5e1
    ✓ Invoke chaincode (102ms)
[03-05-2019 21:44:57] [fabric/fabric-interface.js] [DEBUG]:     User testESY51C4NLMtXTjvzuVH2BKANdLdmF
[03-05-2019 21:44:57] [fabric/user-utils.js] [DEBUG]:    Found user : testESY51C4NLMtXTjvzuVH2BKANdLdmF
[03-05-2019 21:44:57] [fabric/fabric-interface.js] [DEBUG]:     Got user :testESY51C4NLMtXTjvzuVH2BKAN
[03-05-2019 21:44:57] [fabric/fabric-interface.js] [DEBUG]:     Set user context: testESY51C4NLMtXTjvz
[03-05-2019 21:44:57] [fabric/fabric-interface.js] [DEBUG]:     Query : RetrieveResident,v1,defaultcc
[03-05-2019 21:44:57] [fabric/fab-utils.js] [INFO]:      Query on Channel:{"name":"foo","orderers":["Or
[03-05-2019 21:44:57] [fabric/fab-utils.js] [DEBUG]:     query request :{"chaincodeId":"defaultcc","fcn
,181,217,34,230,227,22,106,225,95,252,112,161,251,79,108,10,15,68]},"_transaction_id":"d2d429d80a40b96
[03-05-2019 21:44:57] [fabric/fab-utils.js] [DEBUG]:     Response from:Error: 2 UNKNOWN: chaincode erro
[03-05-2019 21:44:57] [fabric/fab-utils.js] [ERROR]:     Payload error
[03-05-2019 21:44:57] [fabric/fab-utils.js] [ERROR]:     Error: 2 UNKNOWN: chaincode error (status: 500
[03-05-2019 21:44:57] [fabric/fabric-interface.js] [ERROR]:     query error : Error:     Error: 2 UNKNO
(node:24164) UnhandledPromiseRejectionWarning: Unhandled promise rejection (rejection id: 1): Error:
(node:24164) [DEP0018] DeprecationWarning: Unhandled promise rejections are deprecated. In the future,
```

Delete the hfc-cvs and hfc-kvs folders inside ~/fab-node-express, delete node_modules folder, clean up the docker containers:

```
cd ~/Fabric/fabric-network/networkup/docker
./fabric.sh down && ./fabric.sh clean
./fabric.sh upAll
```

Verify that nodejs version 8.9.0 and docker containers are up and running

```
dockerps
node --version
```

Install node modules (WITH VERSION 8.9.0)

```
cd ~/Fabric/fab-node-express
```

npm install

Rerun the test

npm run test

Once the tests pass, start the server.

npm run dev

You can navigate to the html rendered by the server:

localhost:3000

As well as the database

localhost:5984

For a web interface I would recommend building a frontend client (Angular) instead of using the server to render. All that needs to be done is to pull the data for specific clients.

Open up a second terminal to run the python script(s)

Press ctl+alt+t:

cd ~/Fabric/fab-node-express

python3 PYTHON_SCRIPT.py arg1 arg2 arg3…..