#include “master.h”

void master\_init(struct Master \*master, int port\_no) {

master->listen\_sd = socket(AF\_INET, SOCK\_STREAM, 0);  
master->address.sin\_family = AF\_INET;  
master->address.sin\_addr.s\_addr = inet\_addr("127.0.0.1");  
master->address.sin\_port = htons(port\_no);  
  
bind(master->listen\_sd, (struct sockaddr\*)&master->address, sizeof(master->address));  
  
listen(master->listen\_sd, 5);  
  
// master\_address.sin\_family = AF\_INET;  
// master\_address.  
  
printf("Master Listening At Port %d\n", port\_no);

}

void master\_listen(struct Master \*master) { master->node\_size = sizeof(master->node\_size);

master->connect\_sd = accept(master->listen\_sd,   
 (struct sockaddr\*)& master->node\_address, &master->node\_size);

}

void master\_process\_incoming\_connection(struct Master \*master) { // Listens to incomming node // Takes details from node // - Node Name // - Subscriber / Publisher // - Topic Name // - Data Type

struct MasterMessage message;  
int flag = read(master->connect\_sd, &message, sizeof(message));  
  
print\_master\_message(message);  
  
  
exit(0);

}

void master\_close\_connection(struct Master \*master) { close(master->connect\_sd); }

void master\_close(struct Master \*master) {

printf("Master Shutting Down\n");

}

#pragma once #include <stdio.h> #include <stdlib.h> #include <unistd.h>

#include <sys/types.h> #include <sys/socket.h>

#include <arpa/inet.h> #include <netinet/in.h>

#include “struct.h” #include “../messages/master.h” #include “../messages/print.h”

void master\_init(struct Master \*master, int port\_no);

void master\_listen(struct Master \*master);

void master\_process\_incoming\_connection(struct Master \*master);

void master\_close\_connection(struct Master \*master);

void master\_close(struct Master \*master);

#pragma once

struct Master{

int listen\_sd, connect\_sd;  
  
struct sockaddr\_in address;  
  
// Incoming Node  
int node\_size;  
struct sockaddr\_in node\_address;

};

#pragma once

#include “../types/node.h”

struct MasterMessage { char node\_name[50]; enum NodeType node\_type; };

#include “node.h”

void init\_node(struct NodeHandle \*nh, char name[]) { printf(“Method Currently Does Nothing”);

nh->socket\_descriptor = socket(AF\_INET, SOCK\_STREAM, 0);  
  
nh->address.sin\_family = AF\_INET;  
nh->address.sin\_addr.s\_addr = inet\_addr("127.0.0.1");  
nh->address.sin\_port = htons((uint16\_t)80808);  
  
  
int result = connect(nh->socket\_descriptor,   
 (struct sockaddr\*)&nh->address,   
 sizeof(nh->address));  
  
if (result == -1)  
{  
 printf("Failed To Connect To CROSS Core\n");  
 exit(0);  
}  
  
struct MasterMessage message;  
  
strcpy(message.node\_name, "BatMan");  
message.node\_type = SUBSCRIBER;  
  
write(nh->socket\_descriptor, &message, sizeof(message));

}

#include <stdio.h> #include <stdlib.h> #include <unistd.h> #include <string.h>

#include <netinet/in.h> #include <arpa/inet.h>

#include “node\_handle.h” #include “../messages/master.h”

void init\_node(struct NodeHandle \*nh, char name[]);

struct NodeHandle{ // int is\_registered;

int socket\_descriptor;  
  
struct sockaddr\_in address;

}; #pragma once

enum NodeType{

SUBSCRIBER,  
PUBLISHER

}; final: cros-core.c master messages sample-publisher gcc -o cros-core.o cros-core.c build/master.o build/messages.o

sample-publisher: sample-publisher.c node gcc -o sample-publisher.o sample-publisher.c build/node.o

master: cross/master/master.c gcc -o build/master.o -c cross/master/master.c

messages: cross/messages/print.c gcc -o build/messages.o -c cross/messages/print.c

node: cross/node/node.c gcc -o build/node.o -c cross/node/node.c

#include<stdio.h>

#include “cross/node/node.h”

int main() { printf(“Hello World”);

struct NodeHandle nh;  
  
init\_node(&nh, "sample-publisher");  
  
return 0;

}