Πανεπιστήμιο Θεσσαλίας - Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

ΗΥ437 - Αλγόριθμοι CAD II

Εαρινό Εξάμηνο - Ακαδημαϊκό Έτος 2025-2026

1η Εργασία

1/3/2026 εως 15/3/2026

Χ. Σωτηρίου

1 1ο Μέρος

Υλοποιήστε ενα διαδραστικό Γραφικό Περιβάλλον Χρήσης (GUI), το οποίο θα υποστηρίζει:

1. παράθυρο για την απεικόνιση της Φυσικής Διαρύθμισης του Κυκλώματος (Layout), με το ομόνυμο πλαίσιο, το οποίο θα απεικονίζει μια δοκιμαστική διάταξη απο ορθογώνια σχήματα και γραμμές σύνδεσης
2. δυνατότητα μεγένθυσης/απο-μεγένθυσης (zoom-in/zoom-out) περιοχής της διάταξης, χρησιμοποιώντας την ρόδα του ποντικιού (scroll)
3. δυνατότητα κανονικοποίησης της μεγένθυσης/απο-μεγένθυσης, με αριστερό κλίκ του ποντικιού
4. μετακίνηση του ορατού πλαισίου με την χρήση των οριζοντίων και καθέτων scrollbars του παραθύρου απεικόνισης
5. περιορισμό της διαδικασίας απεικόνισης και κλήση διαδικασιών ζωγραφικής βάση του ορατού πλαισίου της οθόνης (clipping)

Σας παρέχεται σχετικός σκελετός κώδικα, που χρησιμοποιεί το `GTK+` API, ο οποίος δημιουργεί την κατάλληλη υποδομή του παραθύρου, μαζί με τις σχετικές οριζόντιες, κάθετες μπάρες για μετακίνηση του ορατού τμήματος. Διαβάστε τα σχόλια του κώδικα και προεκτείνετε τον κατάλληλα για να ολοκληρωθούν οι παραπάνω λειτουργίες.

2 2ο Μέρος

Οι στόχοι του 2ου μέρους είναι (1) να χρησιμοποιήσετε ενα παρεχόμενο API, ώστε να φορτώσετε στην στο εργαλείο σας αρχεία LIB, LEF, DEF kai VERILOG, καθώς και (2) να απεικονίσετε τα στοιχεία του κυκλώματος στην οθόνη. Η τρέχουσα μορφή του C/C++ APIείναι η παρακάτω.

```
// API - file api.h for libpathviz.so //

// *** Function Prototypes *** //

// General Functions //

extern void init_globals();

// Floorplan //

void get_floorplan_dimensions(double *, double *, double *, double *, double *, double *);

// Placement Related //

// Components //

extern unsigned long get_total_components(int);
extern unsigned long get_total_placed_components(int);
extern int get_component_iterator(unsigned long *, int *);
extern int next_component_iterator(unsigned long *, int *);
extern void get_component_name(unsigned long, int, char **);
extern int get_component_is_placed(unsigned long, int);
extern int get_component_location(unsigned long, int, double *, double *);
extern int get_component_dimensions(unsigned long, int, double *, double *);
extern int get_component_CCs(unsigned long, int, unsigned long **, int **, unsigned long *);
extern int get_component_output_gatepins(unsigned long, int, unsigned long **, int **, unsigned long *);
extern int get_gatepin_slack(unsigned long, int, char, double *, double *);

// Top-Level IOs //

extern int get_toplevelio_iterator(unsigned long *);
extern int next_toplevelio_iterator(unsigned long *);
extern void get_toplevelio_name(unsigned long, char **);
extern int get_toplevelio_location(unsigned long, double *, double *);
extern int get_toplevelio_CCs(unsigned long , unsigned long **, int **, unsigned long *);

// Initialise CCs for both Components and Top-Level IOs //

int init_components_toplevelios_CCs();

// TCL Commands Interface //

int load_lef(ClientData, Tcl_Interp *, int, Tcl_Obj *const*);
int load_lib(ClientData, Tcl_Interp *, int, Tcl_Obj *const*);
int load_verilog(ClientData, Tcl_Interp *, int, Tcl_Obj *const*);
int load_def(ClientData, Tcl_Interp *, int, Tcl_Obj *const*);
int io_place(ClientData, Tcl_Interp *, int, Tcl_Obj *const*);
```

Figure 1: api.h (1/2)

```
// *** Externals *** //

extern int initialise_floorplan(ClientData clientdata, Tcl_Interp *interp, int objc, Tcl_Obj *const* objv);
extern int set_component_location(ClientData clientdata, Tcl_Interp *interp, int objc, Tcl_Obj *const* objv);
extern int set_port_location(ClientData clientdata, Tcl_Interp *interp, int objc, Tcl_Obj *const* objv);
extern int create_clock(ClientData clientdata, Tcl_Interp *interp, int objc, Tcl_Obj *const* objv);
extern int set_input_delay(ClientData clientdata, Tcl_Interp *interp, int objc, Tcl_Obj *const* objv);
extern int set_output_delay(ClientData clientdata, Tcl_Interp *interp, int objc, Tcl_Obj *const* objv);
extern int all_inputs(ClientData clientdata, Tcl_Interp *interp, int objc, Tcl_Obj *const* objv);
extern int all_outputs(ClientData clientdata, Tcl_Interp *interp, int objc, Tcl_Obj *const* objv);
extern int get_ports(ClientData clientdata, Tcl_Interp *interp, int objc, Tcl_Obj *const* objv);
extern int compute_delay(ClientData clientdata, Tcl_Interp *interp, int objc, Tcl_Obj *const* objv);
extern int compute_slack(ClientData clientdata, Tcl_Interp *interp, int objc, Tcl_Obj *const* objv);
extern int report_timing(ClientData clientdata, Tcl_Interp *interp, int objc, Tcl_Obj *const* objv);

extern int currentflowstep;
extern int lefloaded;
extern int libloaded;
extern int defloaded[];
extern int verilogloaded[];

// NOTE: USE tclinterp as global variable, to ensure TCL commands may call others //
```

Figure 2: api.h (2/2)

Θα πρέπει να κάνετε compile με την shared βιβλιοθήκη `libpathviz.so` (για λειτουργικό CentOS 7/8/9), που θα κατεβάσετε απο την σελίδα του μαθήματος. Σαν σημείο αναφοράς μπορείτε να κατεβάσετε και το εργαλείο απο το GitHub (κλειστού κώδικα) για να δοκιμάσετε την διαδικασία και να συγκρίνετε το layout. Το σχετικό URL είναι:
`https://github.com/Silicon-Highway-Technologies/Si-Time`.
Παρακάτω φαίνεται πως θα πρέπει να κάνετε compile το πρόγραμμα σας.

```
export LD_LIBRARY_PATH=.

gcc -o <output> <YOUR C FILES> -g `pkg-config --cflags gtk+-2.0`
-lm `pkg-config --libs gtk+-2.0` -lgthread-2.0 -pthread -lglib-2.0
-lreadline -ltcl -L./ -lpathviz
```

Figure 3: compilation script

Αφού φορτώσετε LEF, DEF ή LEF, VERILOG, DEF (δεν χρειάζεται να φορτώσετε LIB σε αυτή την εργασία), θα πρέπει να ζωγραφίσετε την τοποθέτηση του κυκλώματος χρησιμοποιώντας το GUI που υλοποιήσατε στην 1η Εργασία. Το πιο βολικό είναι να χρησιμοποιήσετε ενα TCL front-end απο το οποίο θα καλείτε τις συναρτήσεις του API.

Παρακάτω φαίνεται ενα τέτοιο παράδειγμα.

```
// *** load_lef_TCL *** //
// command: load_lef //
int load_lef_TCL(ClientData clientdata, Tcl_Interp *interp, int objc, Tcl_Obj *const* objv)
{
  load_lef(clientdata, interp, objc, objv);
}
```

Figure 4: TCL API function definition Example

Μπορείτε να ζωγραφίσετε απευθείας το layout, ή όταν ο χρήστης χρησιμοποιήσει κάποια εντολή TCL.

Η προθεσμία παράδοσης της Εργασίας είναι η **15/3/2026**. Μέχρι τότε θα πρέπει να έχετε υποβάλλει τις λύσεις των ασκήσεων μέσω του `e-Class`.