The following code answers the quiz question.

```
#include "pmod_keyboard.h"


ap_uint<8> get_seven_segment_code(ap_uint<5> key_code)
{

	ap_uint<8> code;
	switch(key_code) {
	case 0:
		code = svn_sg_code[0];
		break;
	case 1:
		code = svn_sg_code[1];
		break;
	case 2:
		code = svn_sg_code[2];
		break;
	case 3:
		code = svn_sg_code[3];
		break;
	case 4:
		code = svn_sg_code[4];
		break;
	case 5:
		code = svn_sg_code[5];
		break;
	case 6:
		code = svn_sg_code[6];
		break;
	case 7:
		code = svn_sg_code[7];
		break;
	case 8:
		code = svn_sg_code[8];
		break;
	case 9:
		code = svn_sg_code[9];
		break;
	case 10:
		code = svn_sg_code[10];
		break;
	case 11:
		code = svn_sg_code[11];
		break;
	case 12:
		code = svn_sg_code[12];
		break;
	case 13:
		code = svn_sg_code[13];
		break;
	case 14:
		code = svn_sg_code[14];
		break;
```

```
	case 15:
		code = svn_sg_code[15];
		break;
	default:
		break;
	}


	return code;
}



typedef enum{COL1, COL2, COL3, COL4} keyboard_column;
void pmod_keyboard(
		ap_uint<4>  rows,
		ap_uint<4> &cols,
		bool        input_refresh_signal,
		ap_uint<8> &seven_segment_data,
		ap_uint<4> &seven_segment_enable ) {

#pragma HLS INTERFACE ap_none port=seven_segment_data
#pragma HLS INTERFACE ap_none port=seven_segment_enable
#pragma HLS INTERFACE ap_none port=rows
#pragma HLS INTERFACE ap_none port=cols
#pragma HLS INTERFACE ap_none port=input_rotate_signal
#pragma HLS INTERFACE ap_ctrl_none port=return


	static keyboard_column state      = COL1;
	static ap_uint<4> active_col = 0b1111;

	keyboard_column next_state = state;
	ap_uint<4> next_active_col = active_col;

	ap_uint<5> key_code        = 0;
	bool no_key_pressed        = 0;

	switch(state) {

	case COL1:
		if (input_refresh_signal == 1) {
			next_active_col = 0b1101;
			next_state = COL2;
		} else {
			next_active_col = 0b1110;
			switch(rows) {
			case 0b1111:
				no_key_pressed = 1;
				break;
			case 0b1110:
				key_code = ker_array[0][0];
				break;
			case 0b1101:
				key_code = ker_array[1][0];
```

```
                        break;
                case 0b1011:
                        key_code = ker_array[2][0];
                        break;
                case 0b0111:
                        key_code = ker_array[3][0];
                        break;
                default:

                        break;
                }

                next_state = COL1;
        }
        break;
case COL2:
        if (input_refresh_signal == 1) {
                next_active_col = 0b1011;
                next_state = COL3;
        } else {
                next_active_col = 0b1101;
                switch(rows) {
                case 0b1111:
                        no_key_pressed = 1;
                        break;
                case 0b1110:
                        key_code = ker_array[0][1];
                        break;
                case 0b1101:
                        key_code = ker_array[1][1];
                        break;
                case 0b1011:
                        key_code = ker_array[2][1];
                        break;
                case 0b0111:
                        key_code = ker_array[3][1];
                        break;
                default:

                        break;
                }

                next_state = COL2;
        }
        break;
case COL3:
        if (input_refresh_signal == 1) {
                next_active_col = 0b0111;
                next_state = COL4;
        } else {
                next_active_col = 0b1011;
                switch(rows) {
                case 0b1111:
                        no_key_pressed = 1;
                        break;
```

```
                case 0b1110:
                        key_code = ker_array[0][2];
                        break;
                case 0b1101:
                        key_code = ker_array[1][2];
                        break;
                case 0b1011:
                        key_code = ker_array[2][2];
                        break;
                case 0b0111:
                        key_code = ker_array[3][2];
                        break;
                default:

                        break;
                }

                next_state = COL3;
        }
        break;
case COL4:
        if (input_refresh_signal == 1) {
                next_active_col = 0b1110;
                next_state = COL1;
        } else {
                next_active_col = 0b0111;
                switch(rows) {
                case 0b1111:
                        no_key_pressed = 1;
                        break;
                case 0b1110:
                        key_code = ker_array[0][3];
                        break;
                case 0b1101:
                        key_code = ker_array[1][3];
                        break;
                case 0b1011:
                        key_code = ker_array[2][3];
                        break;
                case 0b0111:
                        key_code = ker_array[3][3];
                        break;
                default:

                        break;
                }

                next_state = COL4;
        }
        break;

default:
        break;
}
```

```
        state      = next_state;
        active_col = next_active_col;

        cols = active_col;
        seven_segment_data   = get_seven_segment_code(key_code);
        seven_segment_enable = ((ap_int<3>)0b111, (bool)no_key_pressed);


}
```