Solar Design Algorithm

The document below defines the calculations used to design and evaluate a PV system in preparation for creating electrical drawings.

The secondary documents are automatically created from this source:

- A printable PDF document describing the algorithm, with no computer code (SDA_standard.pdf).
- Key computer code used in FSEC's online express drawing creation application (SDA.js).
- A printable PDF document describing the algorithm and it's related computer code (SDA.pdf).

Note: For each section, the symbols are pre-pended by a section name to assist with their use in the computer code, in the form of "section.symbol".

Micro Inverter System Calculations

Calculations

Description	Symbol	Calculation	Unit
Maximum source/branch power	source.max_power	module.pmp * array.largest_string	W
Maximum source/branch current	source.current	inverter.max_ac_output_current / 240 * array.largest_string	A
Maximum array power	array.pmp	array.num_of_modules * module.pmp	W

```
source.max_power = module.pmp * array.largest_string;
source.current = inverter.max_ac_output_current / 240 * array.largest_string;
array.pmp = array.num_of_modules * module.pmp;
```

Array checks

The total array power must be less than 10,000W.

```
error_check.power_check_array = array.pmp > 10000;
// If error check is true, flag system design failure, and report notice to user.
if( error_check.power_check_array ){ report_error( 'Array total power exceeds 10kW' );}
```

Branch checks

The largest number of microinverters per branch must not exceed the maximum number allowed by the manufacturer.

error_check.micro_branch_too_many_modules = array.largest_string > inverter.max_unitsperbranch;
// If error check is true, flag system design failure, and report notice to user.
if(error_check.micro_branch_too_many_modules){ report_error('The system has too many inverters per branch circuit.');}

error_check.micro_branch_too_few_modules = array.smallest_string < inverter.min_unitsperbranch;
// If error check is true, flag system design failure, and report notice to user.
if(error_check.micro_branch_too_few_modules){ report_error('The system has too many inverters per branch circuit.');}

The total nominal module power output for each branch must not exceed the manufacturer's limit.

error_check.micro_branch_too_much_power = source.max_power > inverter.max_watts_per_branch;
// If error check is true, flag system design failure, and report notice to user.
if(error_check.micro_branch_too_much_power){ report_error('The branch circuit power limit has exceeded the manufacturer's limit.');}

Module - Inverter checks

The module(s) power must be within the inverter manufacturer's limits.

```
error_check.module_power_too_high = module.pmp > inverter.max_panel_wattage;
// If error check is true, flag system design failure, and report notice to user.
if(error_check.module_power_too_high ){ report_error( 'Microinverter is undersized for module.' );}
error_check.module_power_too_low = module.pmp < inverter.min_panel_wattage;
// If error check is true, flag system design failure, and report notice to user.
if(error_check.module_power_too_low ){ report_error( 'Microinverter is oversized for module.' );}
```

The module's operating voltage must be less than the inverter maximum operating voltage.

```
error_check.module_voltage = module.vmp > inverter.vmax;
// If error check is true, flag system design failure, and report notice to user.
if(error_check.module_voltage ){ report_error( 'Module voltage exceeds inverter maximum.' );}
```

The selected module can not have more cells than allowed by the inveter manufacturer.
error_check.module_cells = module.total_number_cells > inverter.max_module_cells ;
// If error check is true, flag system design failure, and report notice to user.
if(error_check.module_cells){ report_error('Module cell count exceeds the maximum allowed by the inverter.');}

Inverter

If max_ac_ocpd is not provided by the manufacturer, it is calculated as follows:

AC_OCPD_max = max_ac_output_current * 1.25

```
inverter.AC_OCPD_max = sf.if( sf.not( inverter.max_ac_ocpd ), inverter.max_ac_output_current * 1.25, inverter.max_ac_ocpd );
```

The nominal_ac_output_power is selected from fields based on the user selected grid voltage. As an example, if the user selects 240 VAC, then:

nominal_ac_output_power = nominal_ac_output_power_240
max_ac_output_current = max_ac_ouput_current_240

```
inverter.nominal_ac_output_power = inverter['nominal_ac_output_power_'+inverter.grid_voltage];
inverter.max_ac_output_current = inverter['max_ac_ouput_current_'+inverter.grid_voltage];
```

Conductor and conduit schedule

For string inverters, these are the circuit names:

- PV Microinverter AC sources

  var circuit_names = [
  'PV Microinverter AC sources',
  //'Combined AC sources',
  ];
  circuit_names.forEach(function(circuit_name){
  circuits[circuit_name] = {};
  });

The maximum current and voltage for the array DC circuits are equal to source.isc and source.voc.

```
circuits['PV Microinverter AC sources'].max_current = source.current;
```

The AC grid voltage is defined by system specifications (user input).

```
circuits['PV Microinverter AC sources'].max_voltage = inverter.grid_voltage;
```

The number of AC conductors is defined by the conductors required by that AC voltage, multiplied by the number of branches in the array.
Total conductors adds one more for the ground.

```
var conductors_options = {
  '120V': ['ground','neutral','L1'],
  '240V': ['ground','neutral','L1','L2'],
  '208V': ['ground','neutral','L1','L2'],
  '277V': ['ground','neutral','L1'],
  '480V Wye': ['ground','neutral','L1','L2','L3'],
  '480V Delta': ['ground','L1','L2','L3'],
};
inverter.conductors = conductors_options[inverter.grid_voltage+'V'];
inverter.num_conductors = inverter.conductors.length - 1;

circuits['PV Microinverter AC sources'].total_cc_conductors = array.num_of_strings * inverter.num_conductors;
circuits['PV Microinverter AC sources'].total_conductors = array.num_of_strings * inverter.num_conductors + 1;
```

For each circuit, calculate the following.

```
circuit_names.forEach(function(circuit_name, i){
  var circuit = circuits[circuit_name];
  circuit.id = i;


  circuit.power_type = sf.index( ['AC', 'AC', 'AC'], circuit.id );
  // If temperature adder is not defined, set it to 0 for use in further calculations.
  circuit.temp_adder = sf.if( circuit.temp_adder, circuit.temp_adder, 0 );
```

The array maximum temperature of the array is equal to the 2% maximum temperature at the install location, or nearest weather station.
For a state wide design, the largest maximum temperature for the state is used.
Rooftop array circuits also have a temperature adjustment defined above.

```
  circuit.max_conductor_temp = array.max_temp + circuit.temp_adder;
  // Use Table 2, lookup: circuit.max_conductor_temp, return the first column.
  circuit.temp_correction_factor = sf.lookup( circuit.max_conductor_temp, tables[2] );
  // Use Table 3, lookup: circuit.total_cc_conductors, return the first column.
  circuit.conductors_adj_factor = sf.lookup( circuit.total_cc_conductors , tables[3] );
```

Minimum required current is 1.25 times the cicuit's max current:

```
  circuit.min_req_OCPD_current = circuit.max_current * 1.25;
```

For strings per MPP tracker of 2 or less, or for inverters with built in OCPD, additional DC OCPD is not required. The AC circuits do require OCPD at the panel.

```
  circuit.OCPD_required = sf.index( [ true ], circuit.id );
  circuit.ocpd_type = sf.index( ['Circuit Breaker'], circuit.id );
```

Choose the OCPD that is greater or equal to the minimum required current.

```
  // Use Table 9, lookup: circuit.min_req_OCPD_current, find the next highest or matching value, return the index column.
  circuit.OCPD = sf.lookup( circuit.min_req_OCPD_current, tables[9], 0, true, true);
  if( circuit.OCPD > 20 ){  circuit.OCPD = 20 }
  if( circuit_name === 'PV Microinverter AC sources' ){ inverter.OCPD = circuit.OCPD; }
```

Choose the conductor with a current rating that is greater than the OCPD rating from NEC table 310.15(B)(16).
NEC chapter 9 table 8 provides more details on the conductor. For DC circuits, 10 AWG wire is used as a best practice.

```
  circuit.min_req_cond_current = sf.if( circuit.OCPD_required, circuit.OCPD, circuit.min_req_OCPD_current );

  // Use Table 4, lookup: circuit.min_req_cond_current, find the next highest value, return the index column.
  circuit.conductor_current = sf.lookup( circuit.min_req_cond_current, tables[4], 0, true);
  // Use Table 4, lookup: circuit.conductor_current, return the first column.
  circuit.conductor_size_min = sf.lookup( circuit.conductor_current, tables[4] );
  if( circuit.conductor_size_min > 10 ){
    circuit.conductor_size_min = 10;
  }
  // Use Table 5, lookup: circuit.conductor_size_min, return the first column.
  circuit.conductor_current = sf.lookup( circuit.conductor_size_min, tables[5], 1);
  // Use Table 6, lookup: circuit.conductor_size_min, return the first column.
  circuit.conductor_strands = sf.lookup( circuit.conductor_size_min, tables[6], 1 );
  // Use Table 6, lookup: circuit.conductor_size_min, return the second column.
  circuit.conductor_diameter = sf.lookup( circuit.conductor_size_min, tables[6], 2 );
  circuit.min_req_conduit_area_40 = circuit.total_conductors * ( 0.25 * PI() * math.pow(circuit.conductor_diameter, 2) );
```

The NEC article 352 and 358 tables are used to find a conduit with a sufficient 40% fill rate to hold the total conductor size for all the conductors.

```
  // Use Table 7, lookup: circuit.min_req_conduit_area_40, find the next highest value, return the first column.
  // circuit.min_conduit_size_PVC_80 = sf.lookup( circuit.min_req_conduit_area_40, tables[7], 1, true );
  // Use Table 8, lookup: circuit.min_req_conduit_area_40, find the next highest value, return the first column.
  circuit.min_conduit_size_EMT = sf.lookup( circuit.min_req_conduit_area_40, tables[8], 1, true );
  circuit.min_conduit_size = circuit.min_conduit_size_EMT;
```

Select further wire details based on code requirements and best practices.

PV Microinverter AC sources:

- Conductor: 'DC+/DC-, EGC'
- Location: 'Conduit/Exterior'
- Material: 'CU'
- Type: 'THWN-2'
- Volt rating: 600
- Wet temp rating: 90
- Conduit type: 'Metallic'

```
  circuit.conductor = sf.index( ['L1/L2, N, EGC', 'L1/L2, N, EGC'], circuit.id );
  circuit.location = sf.index( ['Conduit/Interior', 'Conduit/Interior'], circuit.id );
  circuit.material = 'CU';
  circuit.type = sf.index( ['THWN-2', 'THWN-2'], circuit.id );
  circuit.volt_rating = 600;
  circuit.wet_temp_rating = 90;
  circuit.conduit_type = sf.index( ['Metallic', 'Metallic'], circuit.id );



  ///////
  // cleanup for display
  if( ! circuit.OCPD_required ){
    circuit.ocpd_type = '-';
    circuit.OCPD = '-';
  }
  circuit.conductor_size_min = circuit.conductor_size_min + ', ' + circuit.conductor_size_min;
  //////

});
```

Interconnection

At least one of the following checks must not fail:

- The sum of 125 percent of the inverter(s) output circuit current and the rating of the overcurrent device protecting the busbar exceeded the ampacity of the busbar.
- The sum of 125 percent of the inverter(s) output circuit current and the rating of the overcurrent device protecting the busbar exceeded 120 percent of the ampacity of the busbar.
- The sum of the ampere ratings of all overcurrent devices on panelboards exceeded the ampacity of the busbar.

```
interconnection.inverter_output_cur_sum = inverter.max_ac_output_current * array.num_of_strings;
interconnection.inverter_ocpd_dev_sum = inverter.OCPD;

interconnection.check_1 = ( ( interconnection.inverter_output_cur_sum * 1.25 ) + interconnection.supply_ocpd_rating ) >
interconnection.bussbar_rating;
interconnection.check_2 = ( interconnection.inverter_output_cur_sum * 1.25 ) + interconnection.supply_ocpd_rating >
interconnection.bussbar_rating * 1.2;
interconnection.check_3 = ( interconnection.inverter_ocpd_dev_sum + interconnection.load_breaker_total ) >
interconnection.bussbar_rating;

error_check.interconnection_bus_pass = sf.and( interconnection.check_1, interconnection.check_2, interconnection.check_3 );
// If error check is true, flag system design failure, and report notice to user.
if( error_check.interconnection_bus_pass ){ report_error( 'The busbar is not compliant.' );}
```

The panel's main OCPD must not exceed the bussbar rating.

```
error_check.interconnection_check_4 = interconnection.supply_ocpd_rating > interconnection.bussbar_rating;
// If error check is true, flag system design failure, and report notice to user.
if( error_check.interconnection_check_4 ){ report_error( 'The rating of the overcurrent device protecting the busbar exceeds the
rating of the busbar. ' );}
```