A generalized state realization of a code \( C \subseteq A \) is defined by a set of state variables \( \{ S_j, j \in I_S \} \), and a set of constraint codes \( \{ C_i, i \in I_C \} \), where \( I_S \) and \( I_C \) are two further discrete index sets. Each state variable \( S_j \) is a finite group, or in the linear case a vector space over \( \mathbb{F} \). Each constraint code \( C_i \) is a group or linear code involving certain subsets of the symbol and state variables. The full behavior of the realization is the set \( \mathcal{B} = (a, s) \) of all configurations of symbol variables \( a \in A \) and state variables \( s \in S = \Pi_{j \in I_S} S_j \) such that all constraints are satisfied. The code generated by the realization is the projection \( \mathcal{C} = \mathcal{B}|_A \) of \( \mathcal{B} \) onto \( A \); i.e., the set of all symbol configurations \( a \in A \) that appear in some \( (a, s) \in \mathcal{B} \).

For example, in a conventional state realization of a linear code \( C \) over a finite field \( \mathbb{F} \), the symbol index set \( I_A \) is a conventional discrete time axis, namely the set of integers \( \mathbb{Z} \), or a subinterval of \( \mathbb{Z} \). The state index set \( I_S \) may be thought of as the set of times that occur between consecutive pairs of times in \( I_A \), and the state time preceding symbol time \( k \in I_A \) is conventionally also denoted by \( k \in I_S \). The constraint codes \( \{ C_k, k \in I_A \} \) are linear codes indexed by the symbol index set \( I_A \), and specify the set of all valid \( (s_k, a_k, s_{k+1}) \) transitions; i.e., for each \( k \in I_A \), \( C_k \) is a subspace of the Cartesian product vector space \( S_k \times A_k \times S_{k+1} \). The full behavior \( \mathcal{B} \) of the realization is the set of all symbol/state trajectories \( (a, s) \) such that \( (s_k, a_k, s_{k+1}) \) is a valid transition in \( C_k \) for all \( k \in I_A \). The code \( \mathcal{C} \) generated by the realization is the set of all symbol trajectories \( a \) that appear in some \( (a, s) \in \mathcal{B} \).

A normal realization is defined as a generalized state realization in which every symbol variable is involved in precisely one constraint code, and every state variable is involved in precisely two constraint codes. Thus a conventional state realization is normal. It is shown in [3] that any generalized state realization may be straightforwardly converted to a normal realization by introducing replication constraints, without essentially increasing the complexity of the realization.

A normal realization has a natural graphical model, in which each constraint code \( C_i \) corresponds to a vertex, each state variable \( S_j \) (which by definition is involved in two constraints) corresponds to an edge connecting the two corresponding constraint vertices, and each symbol variable \( A_k \) (which by definition is involved in one constraint) corresponds to a leaf or “half-edge” connected to the corresponding constraint vertex.

For example, Figure 1 shows the graph corresponding to a conventional state realization, which is a simple chain graph. Here vertices are represented by square boxes, and the “half-
III. DUAL NORMAL REALIZATIONS

The central duality result of [3] is the following: given a normal realization of a code \( C \), the dual normal realization generates the dual code \( C^\perp \). For simplicity of exposition, we will explain this result only for the case where \( C \) is a linear code over a finite field \( F \), but it holds also in the group case; see [3]. In the linear case, the dual code \( C^\perp \) is the usual orthogonal code to \( C \) under the usual symbolwise inner product.

We have seen that a normal realization for \( C \) is defined by a set of symbol variables \( \{A_k, k \in I_A\} \), a set of state variables \( \{S_j, j \in I_S\} \), and a set of constraint codes \( \{C_i, i \in I_C\} \), where each symbol variable is involved in one constraint code, and each state variable is involved in two constraint codes.

The definition of a dual normal realization is slightly simpler in the case of a linear code \( C \) over the binary field \( \mathbb{F}_2 \) than in the general case, so we discuss the binary case first. Then the dual normal realization is defined by the same sets of symbol and state variables, and by the set of orthogonal constraint codes \( \{C^\perp_i, i \in I_C\} \), each involving the same variables as in the primal realization. The graph of the dual realization is thus the same as the graph of the primal realization, except that each constraint code \( C_i \) is replaced by its orthogonal code \( C^\perp_i \).

**Example 1.** Consider the rate-1/2 binary linear time-invariant convolutional code \( C \) generated by the degree-2 generators \((1 + D^2, 1 + D + D^2)\), in standard \( D \)-transform notation. In other words, \( C \) is the set of all output sequences of the single-input, two-output linear time-invariant system over \( \mathbb{F}_2 \) whose impulse response is \((11, 01, 11, 00, \ldots)\). This system has a conventional four-state realization as in Figure 1 in which each symbol variable \( A_k \) may be taken as \((\mathbb{F}_2)^2\), each state variable \( S_k \) may also be taken as \((\mathbb{F}_2)^2\), and each constraint code \( C_k \) is the \((6, 3)\) binary block code generated by the three generators

\[
\begin{array}{c|ccc}
00 & 11 & 10; \\
10 & 01 & 01; \\
01 & 11 & 00,
\end{array}
\]

which represent the three nontrivial (state, symbol, next-state) transitions in the impulse response of the system. The orthogonal code \( C^\perp_k \) may easily be seen to be the \((6, 3)\) binary block code generated by the three generators

\[
\begin{array}{c|ccc}
00 & 11 & 01; \\
01 & 10 & 10; \\
10 & 11 & 00,
\end{array}
\]

which represent the three nontrivial, (state, symbol, next-state) transitions in the impulse response of \( C^\perp \).

For a linear code \( C \) over a nonbinary field \( F \), one further trick (originally introduced by Mittelholzer [6] to dualize conventional state realizations over groups) is needed to define the dual normal realization: namely, in terms of the graph of the realization, insert a sign inverter in the middle of every edge. In other words, invert the sign of each state variable \( S_k \) in one of the two constraint codes in which it is involved. This is illustrated in Figure 2 for a conventional state realization.

**Example 2 (cf. [1], [2]).** Consider the rate-2/3 linear time-invariant convolutional code \( C \) over \( \mathbb{F}_3 \) with \( g_1(D) = (1 + D^2, 2 + D, 0) \) and \( g_2(D) = (1, 0, 2) \). In other words, \( C \) is the set of all output sequences of the two-input, three-output linear time-invariant system over \( \mathbb{F}_3 \) whose impulse responses are \((120, 010, 100, 000, \ldots)\) and \((102, 000, \ldots)\). This system has a conventional nine-state realization as in Figure 1 in which each symbol variable \( A_k \) may be taken as \((\mathbb{F}_3)^2\), each state variable \( S_k \) may be taken as \((\mathbb{F}_3)^2\), and each constraint code \( C_k \) is the \((7, 4)\) ternary block code generated by the four generators

\[
\begin{array}{c|ccc}
00 & 120 & 10; \\
10 & 010 & 01; \\
01 & 100 & 00; \\
00 & 102 & 00,
\end{array}
\]

which represent the four nontrivial \((s_k, a_k, s_{k+1})\) transitions in the two impulse responses of the system. The orthogonal code \( C^\perp_k \) is the \((7, 3)\) ternary block code generated by the three generators

\[
\begin{array}{c|ccc}
00 & 010 & 12; \\
21 & 202 & 11; \\
22 & 111 & 00,
\end{array}
\]

which represent the three nontrivial \((s'_k, a'_k, s'_{k+1})\) transitions.
in the impulse response of a conventional state realization of a single-input, three-output linear system over \( \mathbb{F}_3 \), with sign inverers as in Figure 2, whose impulse response is \((010, 202, 111, 000, \ldots) \), or \((2D + D^2, 1 + D^2, 2D + D^2) \) in \( D \)-transform notation. (Note the unconventional basis of the dual state space.) This is indeed the generator of the orthogonal convolutional code \( C^\perp \) under our symbolwise definition of the inner product. (For the more usual sequence-wise definition of the inner product, we need to take the time-reversal of \( C^\perp \), which in this case is the code generated by \((1 + 2D, 1 + D^2, 1 + 2D) \).)

IV. MacWilliams Identities

Given these duality results, various MacWilliams-type identities may be obtained in a more or less standard manner. We follow the development in [4].

Every finite abelian group \( T \) is a direct product of cyclic groups. In particular, every finite field \( F \) has \( q = p^m \) elements for some prime \( p \) and is isomorphic as an additive group to \((\mathbb{Z}_p)^m\), and every vector space over a finite field \( \mathbb{F}_p \) of dimension \( d \) is isomorphic to \((\mathbb{Z}_p)^{md} \). Thus, for some integer \( n \), we may take \( T = (\mathbb{Z}_p)^n \), the set of \( n \)-tuples of elements of \( \mathbb{Z}_p \).

Given a complex-valued function \( x : T \rightarrow \mathbb{C}, t \mapsto x(t) \) defined on \( T = (\mathbb{Z}_p)^n \), its (Fourier) transform is the complex-valued function \( X : F \rightarrow \mathbb{C}, f \mapsto X(f) \) defined on \( F = (\mathbb{Z}_p)^n \) by

\[
X(f) = \sum_T x(t) \omega^{ft}, \quad f \in F,
\]

where \( \omega \) is a primitive complex \( p \)th root of unity, and \( f, t \in \mathbb{Z}_p \) is the ordinary dot product between the \( n \)-tuples \( f, t \in (\mathbb{Z}_p)^n \) and \( t \in (\mathbb{Z}_p)^n \) over \( \mathbb{Z}_p \). If we view \( x = \{ x(t), t \in T \} \) as a vector indexed by \( T \), and similarly \( X = \{ X(f), f \in F \} \) as a vector indexed by \( F \), then the transform can be expressed in matrix form as

\[
X = \mathcal{H}x,
\]

where the transform matrix is \( \mathcal{H} = \{ \omega^{ft}, f, t \in T \} \).

Note that \( \mathcal{H}^T = \mathcal{H} \), where \( \mathcal{H}^T \) denotes the transpose of \( \mathcal{H} \).

From the orthogonality relation

\[
\sum_F \omega^{ft} = \begin{cases} |F|, & t = 0; \\ 0, & t \neq 0, \end{cases}
\]

we obtain the matrix equation

\[
\mathcal{H}^* = |F| I_{|F|},
\]

where \( \mathcal{H}^* = \{ \omega^{-ft}, f, t \in F, t \in T \} \), \( |F| = |T| = p^n \), and \( I_{|F|} \) is the \( |F| \times |F| \) identity matrix. In other words, the inverse of \( \mathcal{H} \) is \( \mathcal{H}^{-1} = |F|^{-1} \mathcal{H}^* \). Thus we obtain the inverse transform

\[
x = \mathcal{H}^{-1} X = \frac{\mathcal{H}^* X}{|F|}.
\]

We say that \( x \) and \( X \) are a transform pair.

We may extend these definitions to a set of indeterminates \( z = \{ z(t), t \in T \} \) indexed by \( T \), rather than a complex-valued function. The transform of this set is then a set of indeterminates \( Z = \{ Z(f), f \in F \} \) indexed by \( F \), where

\[
Z = \mathcal{H}z.
\]

Again, we have the inverse transform relationship

\[
z = \mathcal{H}^{-1}Z = \frac{\mathcal{H}^*Z}{|F|},
\]

and we say that \( z \) and \( Z \) are a transform pair.

For example, if \( T = \mathbb{Z}_2 \), then \( Z(0) = z(0) + z(1) \) and \( Z(1) = z(0) - z(1) \); similarly, \( z(0) = \frac{1}{2}(Z(0) + Z(1)) \), and \( z(1) = \frac{1}{2}(Z(0) - Z(1)) \).

Now let us consider weight enumerators, initially for the case of a conventional state realization over a finite field \( F \) as in Figure 1. We define the complete weight adjacency matrix (CWAM) of each constraint code \( C_k \subseteq S_k \times A_k \times S_{k+1} \) as follows.

If \( A_k = F^n \), then define the complete weight enumerator of the \( n \)-tuple \( a = (a_1, \ldots, a_n) \in F^n \) as the product \( w(a) = \Pi_{1 \leq i \leq n} w(a_i) \), where \( w = \{ w(a), a \in F \} \) is a set of indeterminates indexed by \( F \). Then the CWAM of \( C_k \) is the matrix \( \Lambda(w) = \{ \Lambda(s_k, s_{k+1}) (w), (s_k, s_{k+1}) \in S_k \times S_{k+1} \} \) defined by

\[
\Lambda(s_k, s_{k+1}) (w) = \sum_{C(s_k, s_{k+1})} w(a_k),
\]

where \( C(s_k, s_{k+1}) = \{ a_k \mid (s_k, a_k, s_{k+1}) \in C_k \} \). Thus each entry \( \Lambda(s_k, s_{k+1}) (w) \) is a homogeneous integer polynomial of degree \( n \) in the \( |F| \) indeterminates \( w = \{ w(a), a \in F \} \).

Now let \( y = \{ y(s_k), s_k \in S_k \} \) and \( z = \{ z(s_{k+1}), s_{k+1} \in S_{k+1} \} \) be sets of indeterminates indexed by the state variables \( S_k \) and \( S_{k+1} \), respectively, and define a generating function \( g_{\Lambda(w)}(y, z) \), a polynomial in the sets of indeterminates \( y \) and \( z \), as follows:

\[
g_{\Lambda(w)}(y, z) = y^T \Lambda(w) z = \sum_{S_k \times S_{k+1}} y(s_k) \Lambda(s_k, s_{k+1}) (w) z(s_{k+1}).
\]

From the definition \( C(s_k, s_{k+1}) = \{ a_k \mid (s_k, a_k, s_{k+1}) \in C_k \} \), it follows that

\[
g_{\Lambda(w)}(y, z) = \sum_{(s_k, a_k, s_{k+1}) \in C_k} y(s_k) w(a_k) z(s_{k+1}).
\]

Example 1 (cont.). The constraint code of Example 1 has the eight codewords \( 00|00|00, 00|11|10, 10|01|01, 10|10|11, 01|11|00, 01|00|10, 11|10|01, 11|01|11 \), corresponding to the eight possible (state, symbol, next-state) transitions. Writing \( \{ w_0, w_1 \} \) instead of \( \{ w(0), w(1) \} \), we see that we may write the CWAM of this constraint code in matrix form
The key duality relation for MacWilliams identities is the Poisson summation formula, which says that “the sum of a function over a linear space is equal to the sum of the Fourier transform of the function over the dual space” [5]. For our case, this formula may be stated as follows:

**Poisson summation formula.** Let \( x \) and \( X \) be a transform pair defined on \( T = (\mathbb{Z}/p)^n \) and \( F = (\mathbb{Z}/p)^n \), respectively, and let \( C \) and \( C^\perp \) be orthogonal subgroups of \( T \) and \( F \), respectively. Then

\[
\sum_{t \in C} x(t) = \frac{1}{|C^\perp|} \sum_{f \in C^\perp} X(f).
\]

Now, applying this formula to the equation above for \( g_{\Lambda(w)}(y, z) \), we obtain

\[
g_{\Lambda(w)}(y, z) = \sum_{c_k} y(s_k) w(\alpha_k) z(s_{k+1}) = \frac{1}{|C_k^\perp|} \sum_{c_k} Y(\hat{s}_k) W(\hat{\alpha}_k) Z(-\hat{s}_{k+1}).
\]

Here we use the fact that the transform of a product is the product of their transforms, where \( Y = H_y y, \ W = H_w w \, \) and \( Z = H_z z \). Note that \( W \) is itself a product transform. Also, since the elements of \( H_z \) are \( H_z(s_k, -s_{k+1}) = \omega^{-s_k + s_{k+1}} \), the matrix \( H_z \) is the conjugate of the usual transform matrix over \( S_k \).

If we define the CWAM \( \hat{\Lambda}(W) \) of \( C_k^\perp \) and its generating function \( g_{\hat{\Lambda}(W)}(Y, Z) \) similarly to the analogous quantities for \( C_k \), we then obtain

\[
g_{\hat{\Lambda}(W)}(Y, Z) = \sum_{c_k^\perp} Y(\hat{s}_k) W(\hat{\alpha}_k) Z(-\hat{s}_{k+1}).
\]

Using inverse transforms, we thus obtain

\[
g_{\Lambda(W)}(Y, Z) = |C_k^\perp| g_{\Lambda(w)}(y, z) = |C_k^\perp| g_{\Lambda(\hat{H}_y^{-1}Y, \hat{H}_z^{-1}Z)}.
\]

This MacWilliams identity shows how the generating function for the CWAM \( \hat{\Lambda}(W) \) of \( C_k^\perp \) may be obtained from that for \( C_k \), or vice versa.

Alternatively, since

\[
g_{\Lambda(w)}(Y, Z) = Y^T \hat{\Lambda}(W) Z
\]

and

\[
g_{\Lambda(w)}(y, z) = y^T \Lambda(w) z = Y^T \hat{\Lambda}(w) Y \hat{\Lambda}(w) Z,
\]

we may simply write

\[
\hat{\Lambda}(W) = |C_k^\perp| \hat{H}_y^{-1} \Lambda(H_w^{-1}W) H_z^{-1},
\]

a MacWilliams identity that shows how the CWAM of \( C_k^\perp \) may be obtained from the CWAM of \( C_k \).

**Example 1 (cont.).** Given the CWAM \( \Lambda(w) \) of the constraint code \( C_k \) of Example 1, the CWAM \( \Lambda(W) \) of the orthogonal constraint code \( C_k^\perp \) is given by the matrix equation at the top of the next page, where we have substituted the dual indeterminates \( W_0 \) and \( W_1 \) for \( w_0 + w_1 \) and \( w_0 - w_1 \).

The Hamming weight adjacency matrix (HWAM) \( \Lambda_H \) of a constraint code \( C_k \) is obtained by substituting \( 1 \) for \( w(0) \) and \( w \) for each \( w(a), a \neq 0 \). Thus each element \( \Lambda_H(s_k, s_{k+1})(w) \) becomes a polynomial of degree \( n \) in the single indeterminate \( w \). The dual indeterminates become \( W(0) = 1 + (|F| - 1)w \) and \( W(a) = 1 - w, a \neq 0 \), which scale to \( 1 \) and \( W = (1 - w)/(1 + (|F| - 1)w) \), respectively. Substituting in the above MacWilliams-type identities for CWAMs, we obtain MacWilliams-type identities for HWAMs. This yields the main result of [1], [2].

**Example 2 (cont.).** For a worked-out example of the HWAM \( \Lambda(W) \) of the orthogonal code \( C_k^\perp \) to the constraint code of Example 2, see [2].

Although our development has focussed on conventional state realizations of linear time-invariant convolutional codes, it may be straightforwardly extended to obtain MacWilliams identities for any generalized state realization of any finite abelian group code defined on an arbitrary graph, because constraint code duality holds in the general case.

**V. DUALIZING THE SUM-PRODUCT UPDATE RULE**

Another duality result in [3] is a general method for dualizing the sum-product update rule, which among other things yields the “tanh rule” of APP decoding. The approach of this paper yields a cleaner derivation of this result.

Again, for simplicity we restrict attention to conventional state realizations, in which each constraint code \( C_k \) specifies the state transitions in \( S_k \times A_k \times S_{k+1} \) that can possibly occur. Let the (right-going) message be any real- or complex-valued function \( m_k(s_k, s_{k+1}) \) of the state variable \( S_k \), and let \( f_k = \{ f_k(\alpha_k), \alpha_k \in A_k \} \) be any real- or complex-valued weight function of the symbol variable \( A_k \). Then the sum-product update rule associated with constraint code \( C_k \) is

\[
m_{k+1}(s_{k+1}) = \sum_{c_k} m_k(s_k) f_k(\alpha_k),
\]

where the sums are over states of the constraint code. This equation is the dual to the forward and backward recursions of APP decoding. Note that the dual state representation is fixed as soon as the basis of the primal state space is fixed.
where $C_k(s_{k+1}) = \{(s_k, a_k) \in S_k \times A_k \mid (s_k, a_k, s_{k+1}) \in C_k\}$. In other words, if we define a set of indeterminates $x = \{x(s_k+1), s_{k+1} \in S_{k+1}\}$, then $m_{k+1}(s_{k+1})$ is the coefficient of $x(s_{k+1})$ in the homogeneous degree-1 multivariate generating function $g_{k+1}(x)$ defined by

$$g_{k+1}(x) = m_{k+1}^T x = \sum_{s_{k+1}} m_{k+1}(s_{k+1}) x(s_{k+1}).$$

From the definition of $C_k(s_{k+1})$, it follows that

$$g_{k+1}(x) = \sum_{c_k} m_k(s_k) f_k(a_k) x(s_{k+1}).$$

Using the Poisson summation formula, we obtain

$$g_{k+1}(x) = \sum_{c_k} m_k(s_k) f_k(a_k) x(s_{k+1})$$

$$= \frac{1}{|C_k^+|} \sum_{\hat{c}_k \in C_k^+} M_k(\hat{s}_k) F_k(\hat{a}_k) X(-\hat{s}_{k+1})$$

$$= \frac{\hat{g}_{k+1}(X)}{|C_k^+|},$$

where we again use the fact that the transform of a product is the product of their transforms, and define transformed functions or indeterminates by corresponding capitalized functions or indeterminates. The left side of this equation is the generating function $g_{k+1}(x)$ of the message $\{m_{k+1}\}$, and the right side is (up to scale) the generating function $\hat{g}_{k+1}(X)$ of the message $M_{k+1}$ obtained by performing the sum-product update algorithm for $C_k^+$ upon the message $M_k$ and the weight function $F_k$. Moreover, the messages $m_{k+1}$ and $M_{k+1}$ form a transform pair.

Consequently, we have the following recipe for performing the sum-product update rule for $C_k$:

1) Transform the incoming messages $m_k$ and $f_k$ to $M_k$ and $F_k$;
2) Perform the sum-product update rule for $C_k^+$ to generate an output message $M_{k+1}$;
3) Inverse transform $M_{k+1}$ to obtain the message $m_{k+1}$, up to the scale factor $|C_k^+|$.

Since the complexity of performing the sum-product update rule for $C_k$ is proportional to $|C_k^+|$, this dual computation may be attractive if $|C_k^+| < |C_k|$.